SIEMENS

TIA-GRAPH

SITRAIN
Digital Industry Academy

SIMATIC programming with GRAPH

siemens.com/sitrain

SIEMENS

SITRAIN
Digital Industry Academy

SIMATIC S7 – Programming with GRAPH

TIA-GRAPH Course

Name: _____

Course from: _____ to: _____

Course Leader: _____

Course Location: _____

This curriculum was created for training purposes. Siemens AG does not guarantee the contents.
Reproduction and duplication of this document and utilization and communication of its content is prohibited without our express permission. Any breach or infringement will result in liability for damages.

Copyright © Siemens AG 2020. All rights reserved, especially the right to apply for patents and/or to register other intellectual property rights and to have these granted.

SITRAIN course offer on the Internet: www.siemens.de/sitrain

Course folder version: V16.00.00 (for STEP7 Version 16.0)

1 Engineering Tool GRAPH
2 S7-1500 Training Device
3 Getting Started with GRAPH
4 Interlock, Supervision and Alarms
5 Test Functions in GRAPH
6 Event-driven Actions
7 Branches and Jumps
8 Operating Mode Management
9 Troubleshooting and Diagnostics
10 Structure of Instance DB
11 General Observations about Step Sequences
12 Training and Support
13
14
15

Table of contents 1

1. Engineering Tool GRAPH 1-2

1.1. TIA Portal – Central engineering framework 1-3

1.2. Scope of performance of products 1-4
1.2.1. STEP 7 product line 1-5
1.2.2. Classification of controllers 1-6

1.3. The meaning of GRAPH 1-7

1.4. GRAPH Runtime system 1-8

1.5. The conveyor model as production line 1-9

1.6. If you want to know more 1-10
1.6.1. TIA Portal: Portal and Project views 1-11
1.6.2. Portal view 1-12
1.6.3. Project view 1-13
1.6.4. Project tree 1-14
1.6.5. Task cards 1-15
1.6.6. Inspector window 1-16
1.6.7. Reference project 1-17
1.6.8. Window arrangement in Project view 1-18
1.6.9. Saving/managing/using window layouts 1-19
1.6.10. Window layout of work area 1-20
1.6.11. Keep editor window in foreground (with split work area) 1-21
1.6.12. Undo and Redo 1-22
1.6.13. Saving a project 1-23
1.6.14. Archiving and retrieving a project 1-24
1.6.15. TIA Portal settings: User interface language 1-25
1.6.16. TIA Portal settings: Language, storage location, layout 1-26

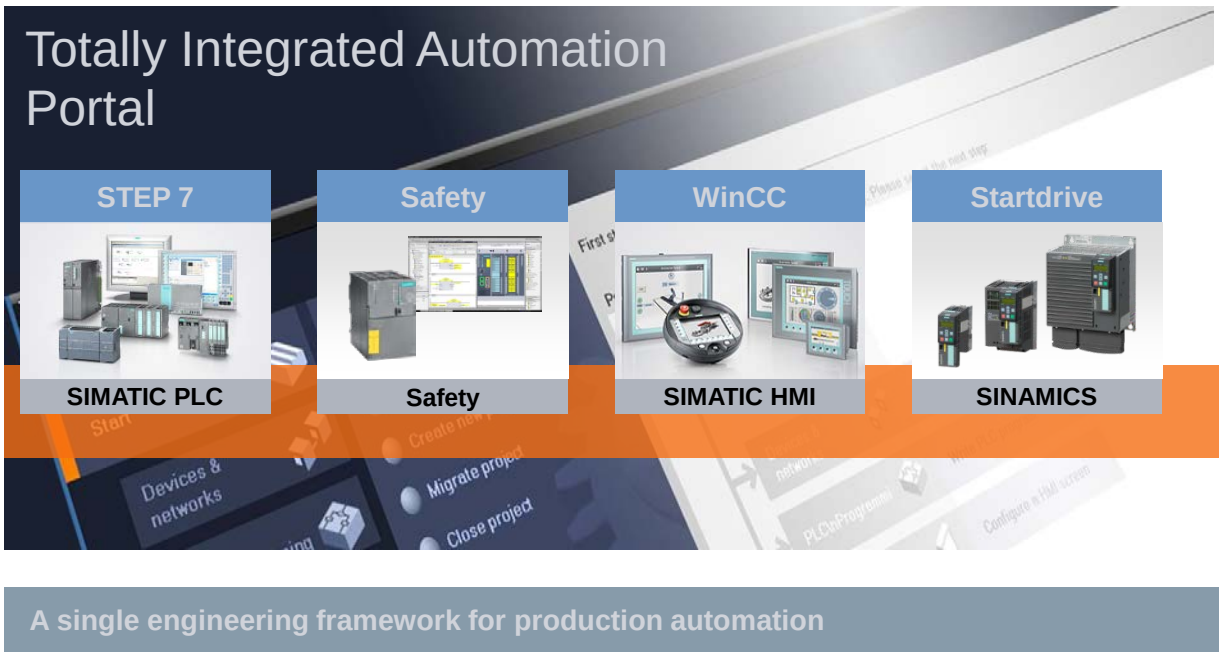
1. Engineering Tool GRAPH

The participant should ...

… receive an overview of the TIA Portal Engineering Framework

... become acquainted with the Graph programming language

1.1. TIA Portal – Central engineering framework



Stand-alone software packages are limited in their consistency and integration.

To achieve full integration and consistency of individual products, a single work environment – an **engineering framework** – is needed.

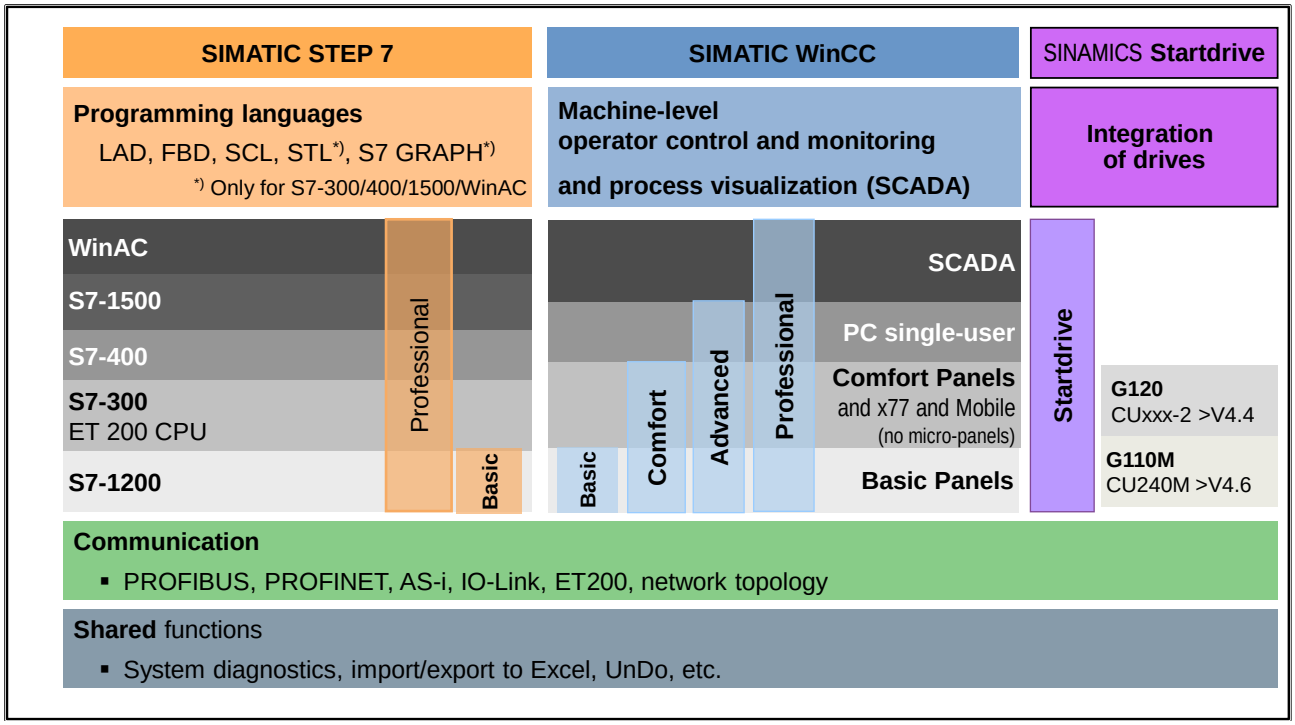
Advantages of a central engineering framework

- Uniform operating concept across all automation tasks with shared services (e.g. configuration, communication, diagnostics)
- Automatic data and project consistency
- Efficient libraries of all automation objects

The most important engineering products are:

- SIMATIC STEP 7
  for programming the PLC
- SIMATIC Safety
  for programming fail-safe PLCs
- SIMATIC HMI
  for configuring the process visualization
- Startdrive
  for parameter assignment of the drives

1.2. Scope of performance of products

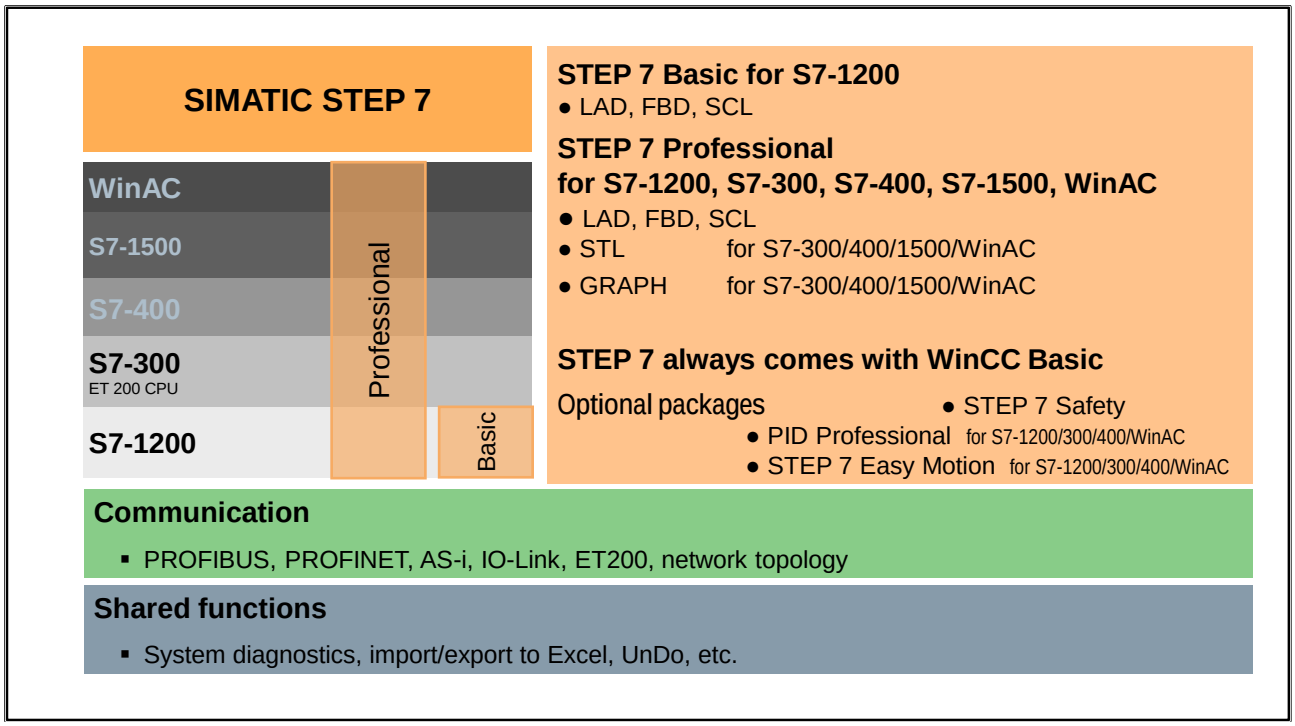


TIA Portal

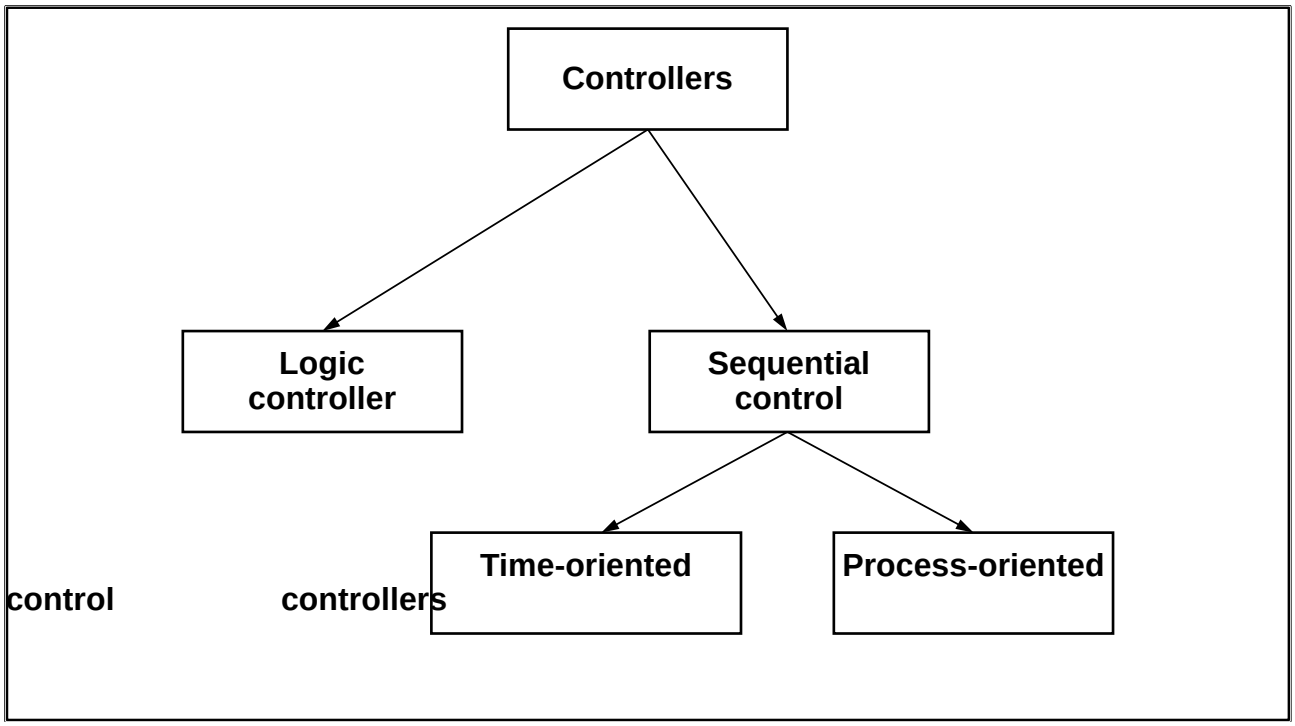
The Totally Integrated Automation Portal is the working environment for integrated engineering with SIMATIC STEP 7 and SIMATIC WinCC.

- Central engineering framework
- Automatic data and project consistency
- Uniform operating concept across all automation tasks
- Efficient libraries of all automation objects

1.2.1. STEP 7 product line



1.2.2. Classification of controllers



Definitions

As the name implies, logic controllers generate the states of outputs by performing logic operations on the inputs. Basic logic functions of Boolean algebra (AND; OR; NOT; XOR) as well as set and reset functions are mainly used for this.

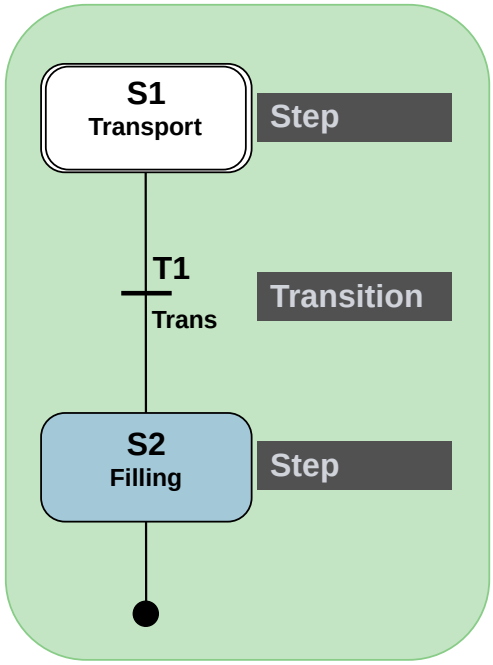
Programming languages such as Ladder Diagram and Function Block Diagram (LAD and FBD) are used for notation of the relationships between inputs and outputs. A logic controller can be distinguished from a sequential control by the time-invariant, fixed relationship between inputs and outputs. This means that the output state is independent of the state history and is only a function of the inputs and not the time or process status.

Sequential control (Sequential Function Charts – SFCs)

In direct contrast, a sequential control is characterized by the running of individual control operations in a certain order under the control of a so-called sequence processor.

The individual steady states of the sequence are called steps. The change from one step to the next (or to multiple next steps) is called a transition and occurs only after certain criteria are met, which are referred to collectively as a step-enabling condition. The criteria of the step-enabling condition can be either characteristic quantities for describing the process state or time intervals.

1.3. The meaning of GRAPH



Sequential Function Chart (S7-GRAPH) for sequential processes

- Graphic program creation
- Structured program design
  - Time saving of up to 50%*
  - Process description
  - Clear, easy to understand
- Describe time-based sequences
  - Steps
  - Transitions
- Integrated diagnostic functions, e.g. monitoring the maximum execution time of a step
- Certified according to PLCopen Base Level

* Compared to programming of the same functions with KOP, FUP, AWL

GRAPH

GRAPH is a graphic programming language for representing SFCs (Sequential Function Charts) in the S7 environment.
With GRAPH, SIMATIC controllers can be programmed quickly and efficiently for execution of time-based and repeating sequences.

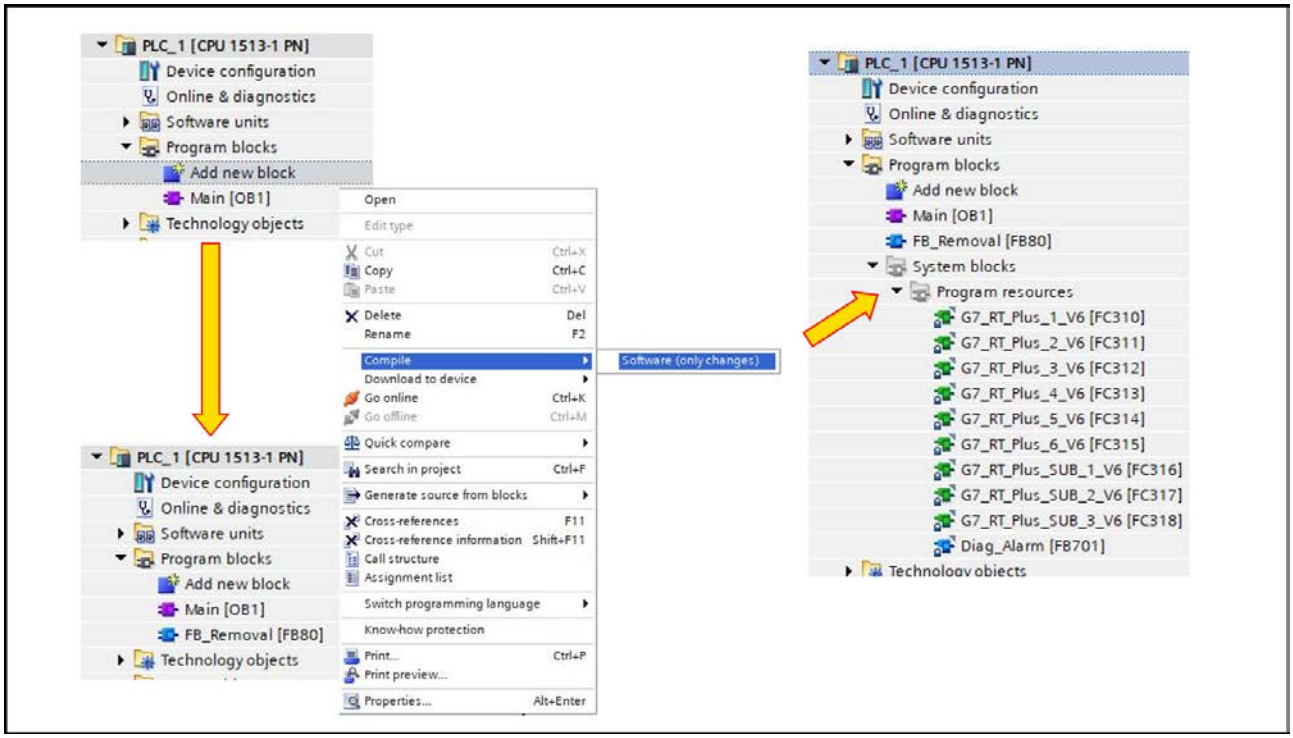
Functionality

- GRAPH enables a structured programming:
- Programming of repeating, time-based sequences
- Defined transitions between process steps
- Jumps, branches, etc.
- Combined with control-specific functions, such as:
  - Bit access to I/O, bit memory, timers, counters, etc.
  - Access to the tag table
  - STEP 7 block access

Advantages of GRAPH

- GRAPH is a straightforward and easy to understand graphic programming language for creating sequential processes.
- GRAPH is an easy to learn programming language. It creates programs that are easy to read (understandable), especially for beginners.
- Easier programming of complex algorithms and processing of complex data structures.
- System integration in the following SIMATIC S7 languages: STL, SCL, FBD and LAD.

1.4. GRAPH Runtime system



General

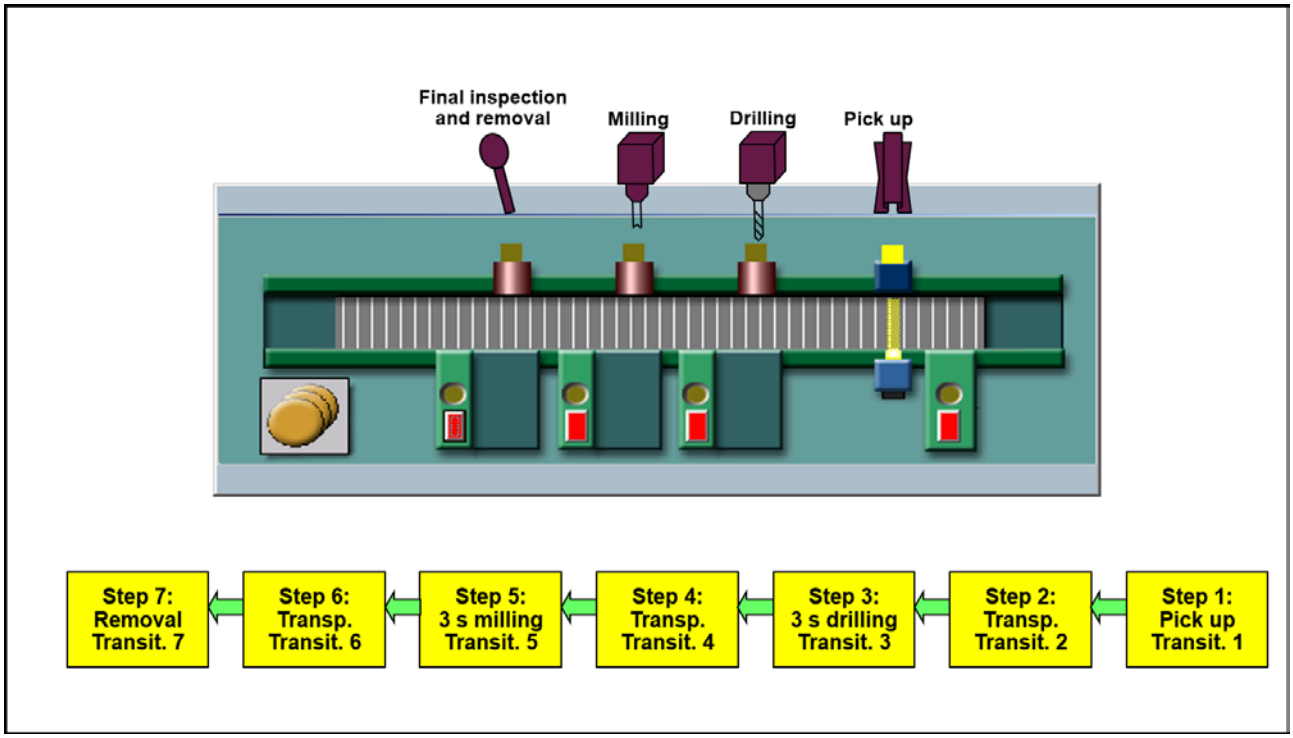
A SIMATIC controller generally executes the user program cyclically. That is, the operating system calls OB1, and its processing time is equal to exactly one cycle, also called the cycle time of the PLC.

In OB1, the user programs the various calls of the blocks, FCs and FBs written by him. The cycle time is extended at a new FB call by the time the controller needs to process the complete block once (top-down principle).

GRAPH

A GRAPH block normally contains a sequential control that consists of one or more steps. The complete processing time of a sequential control lasts much longer than the cycle of a PLC. For this reason, the GRAPH programming language requires its own runtime system for controlling the sequential control. This runtime system is automatically created in the "System blocks" folder after successful compilation of the first GRAPH block and is managed there by the TIA Portal.

1.5. The conveyor model as production line


Final inspection and removal
Milling
Drilling
Pick up
Step 7: Removal Transit. 7
Step 6: Transp. Transit. 6
Step 5: 3 s milling Transit. 5
Step 4: Transp. Transit. 4
Step 3: 3 s drilling Transit. 3
Step 2: Transp. Transit. 2
Step 1: Pick up Transit. 1

General

A control program whose processing steps are executed sequentially in a defined time sequence is referred to as a sequential control.

The control task is broken down into individual steps and programmed as a step sequence (sequencer). Actions for activating the command output are assigned to each step. During execution, the sequencer is processed step by step. Step-enabling conditions or transitions control the transition from one step to the next.

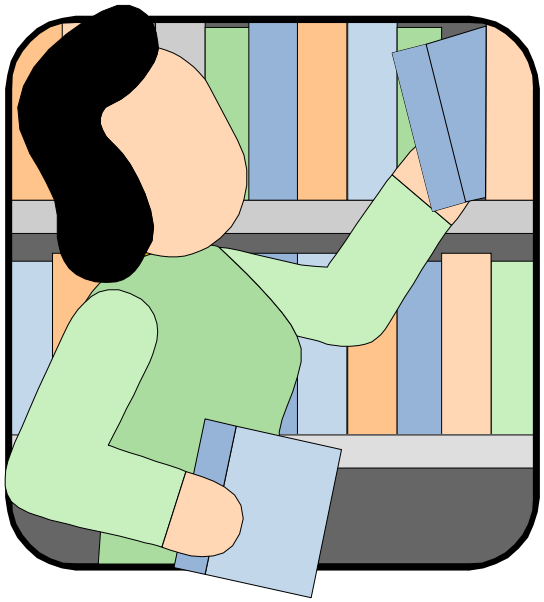
If a step is active, the assigned actions are executed. The next transition is simultaneously activated and its step-enabling conditions are logically queried. If these queries are satisfied, the preceding step is deactivated and the next step is activated.

Practice

The use of a combination of a logic controller and sequential control is common in practice. Operating mode and alarm blocks are generally implemented as logic controllers. The processing operation as such is programmed as a sequential control.

The term "sequential control" thus does not stand for a special operating mode of the PLC, but rather for a type of structuring and programming of the user program. Typical application ential control is advantageous programming as a sequ examples for which
assembly machines, car washes, packaging machines and traffic lights – in other words, processes that were conventionally handled with sequence processors.

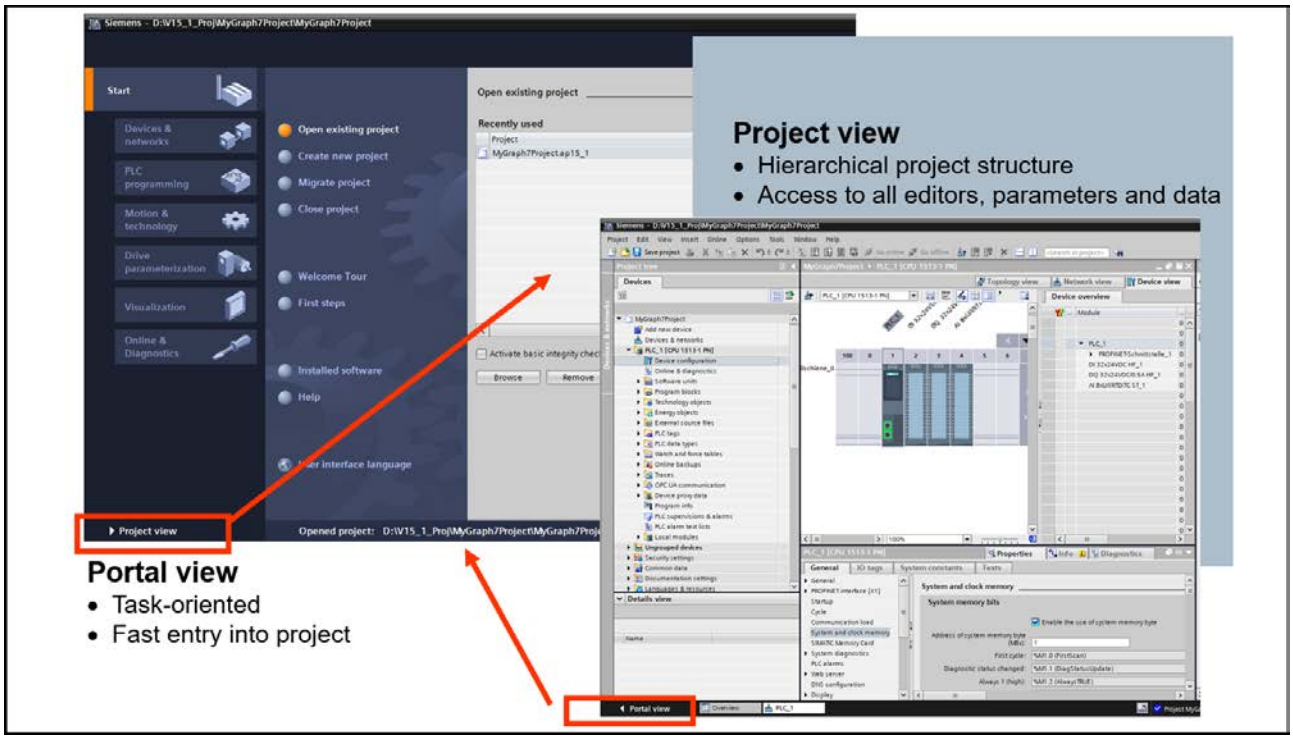
1.6. If you want to know more



Note

The following pages provide additional information or serve as a reference that can be consulted for more complete information on a subject.

1.6.1. TIA Portal: Portal and Project views



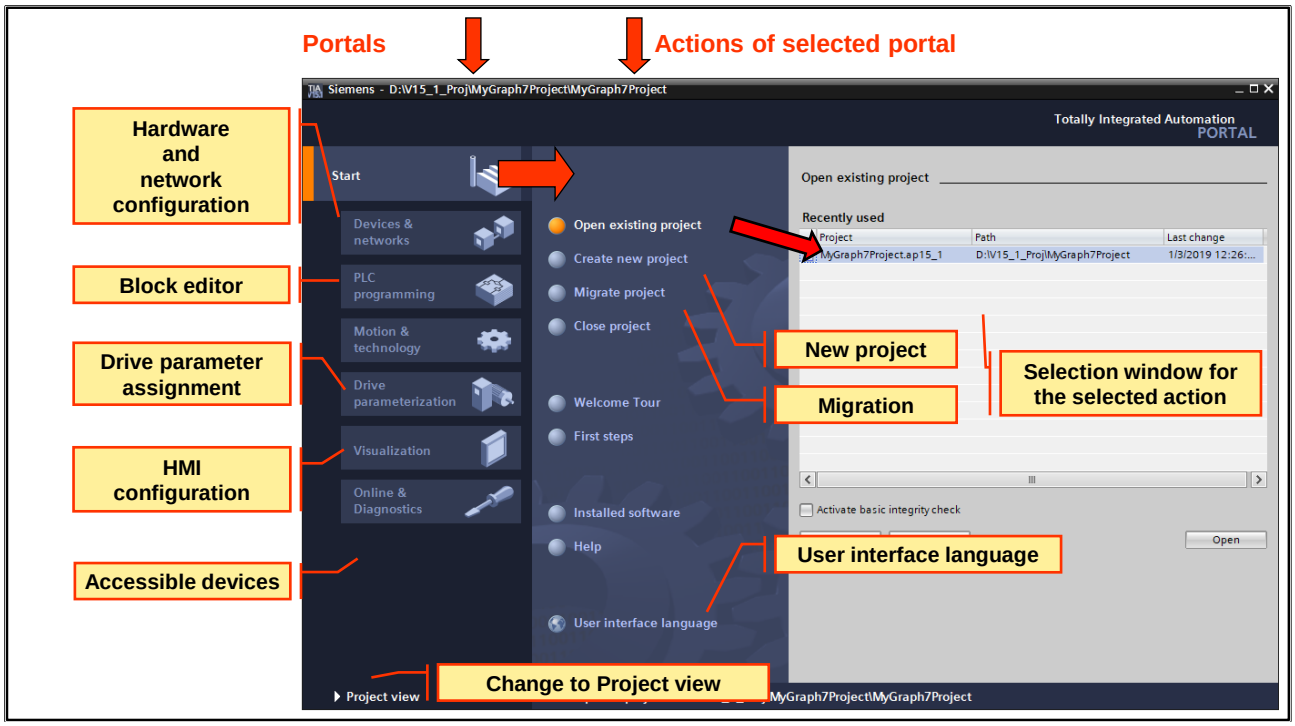
Portal view

- Task-oriented working
- Fast entry into project through guided operation

Project view

- Hierarchical structure of the project
- The required editors open as needed for a task
- All editors, parameters and data are in one view

1.6.2. Portal view



Layout of the portal view:

- Portals for different tasks
- Actions for the selected portal
- Selection window for the selected action

Portals

Access to devices, components and their connections.

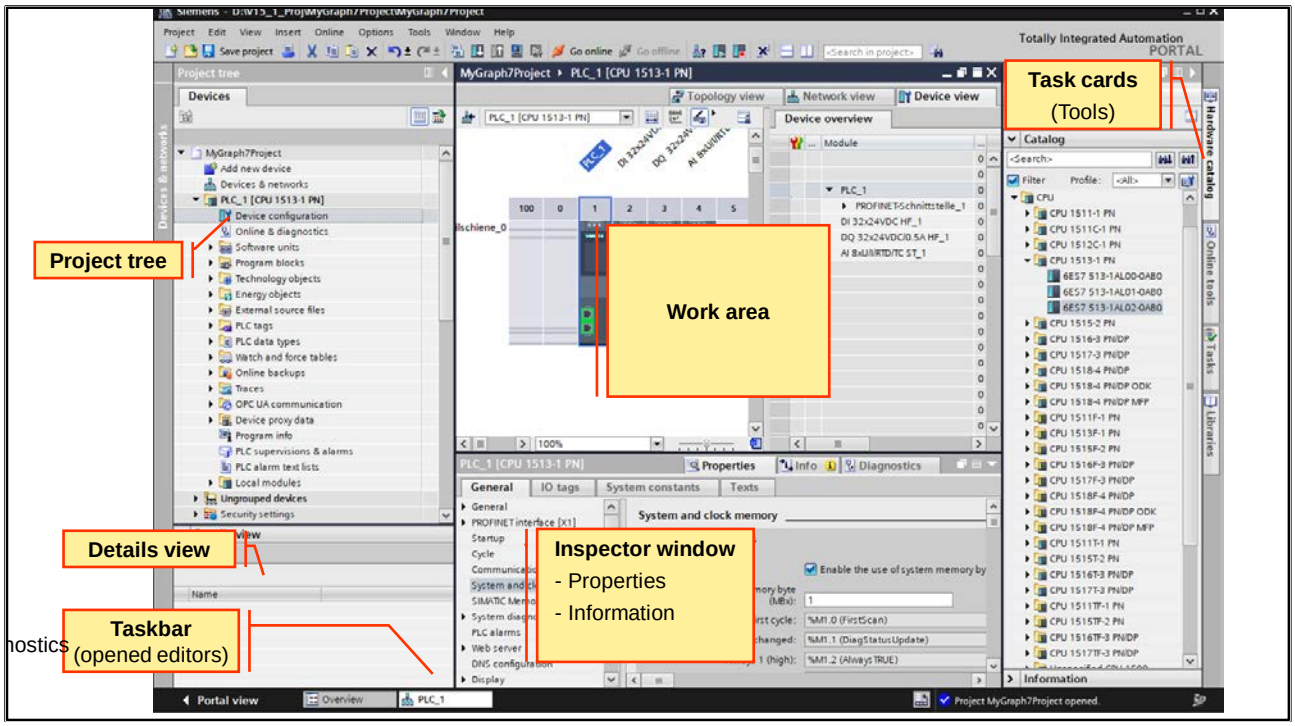
Actions

Depending on the selected portal, the actions that are available for selection are offered here. Context-sensitive help is available to you in every portal.

Selection window

The selection window is available in all portals. The content of the window adapts to your current selection.

1.6.3. Project view



Project tree

The project tree contains all components and project data of an automation solution. These can all be opened from the project tree.

Work area

The objects that are opened for editing purposes are displayed in the work area. These objects include hardware components, blocks, PLC tag table, HMI screens, etc. If multiple objects are opened simultaneously, they are displayed as tabs in the taskbar.

Task cards

These provide tools for configuring/programming. The content of the task cards is dependent on the object that is displayed in the work area.

If a hardware station is open, for example, the hardware catalog is available as a task card. If a program block is open, a task card with instructions is available.

Inspector window

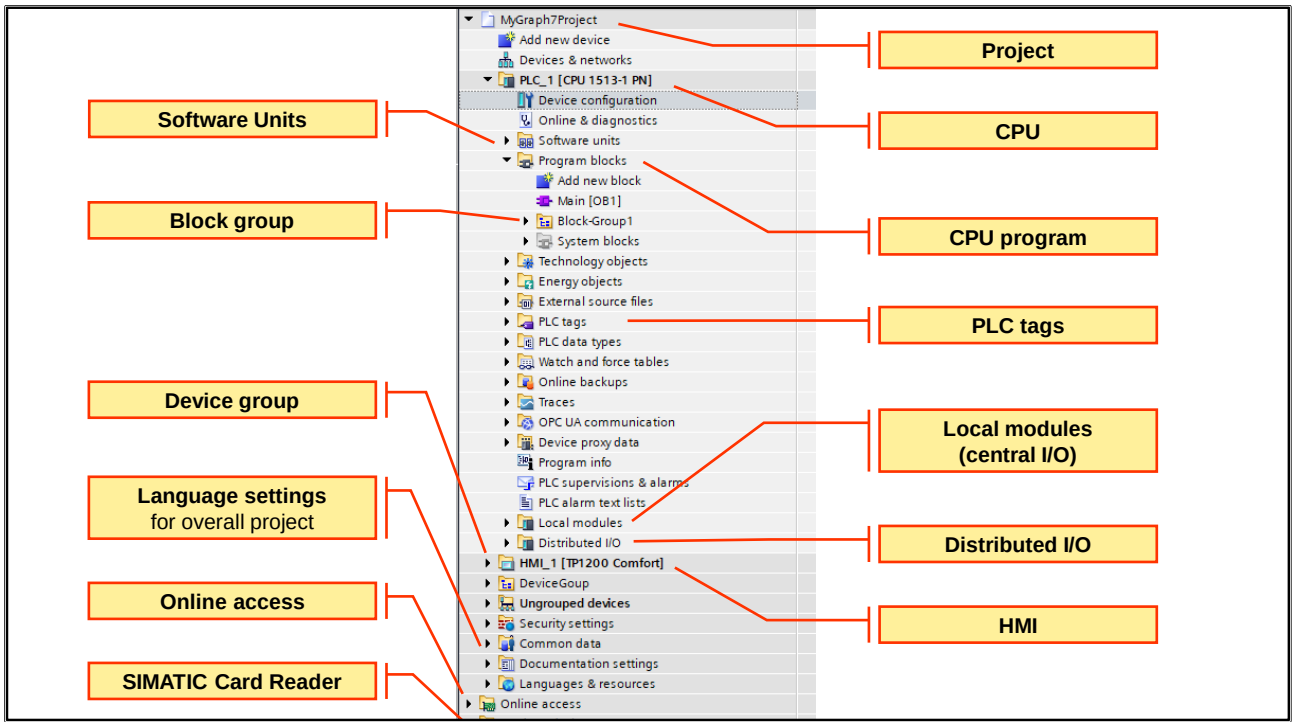
The Inspector window displays additional information on a selected object or on the executed action. The properties of selected objects (e.g. of screens, screen objects, tags) can also be edited there.

All system events of the engineering framework are displayed in the Inspector window, e.g. system events that arise when a project is generated. This window should always be checked for errors and warnings after project generation is complete.

Details view

The details view contains a help window. The elements of the configuration object selected in the project tree are displayed there. These can be used in the active work area (by moving them into the work area using drag-and-drop). This enables fast access to the needed objects (e.g. tags).

1.6.4. Project tree



The "Project tree" window gives you access to all components and project data. All elements and all available objects of a project are displayed in a tree structure in the project tree and can be opened from there with a double click.

The following actions can be performed:

- Add new components (controllers, HMI devices, etc.)
- Edit existing components
- Query and modify properties of existing components
- Diagnose accessible components

Objects (whole stations or blocks of a station) can be organized into groups to make the project tree easier to read.

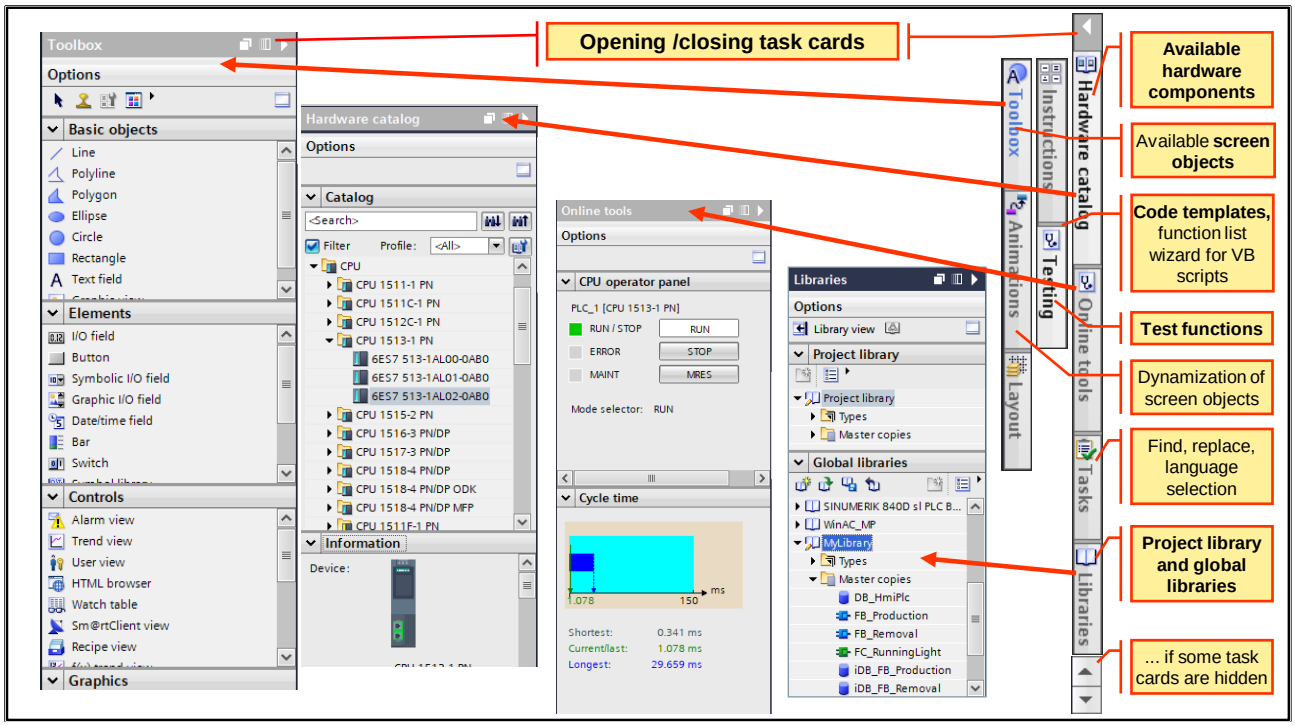
Displaying and hiding the structure section

A subordinate structure is indicated by the black triangle ▶. By clicking the triangle, the subordinate structure level can be displayed ▶→▼ and hidden again ▼→▶.

Details view

Certain contents of a selected object are shown in the Details view. Examples of possible contents are text lists or tags.

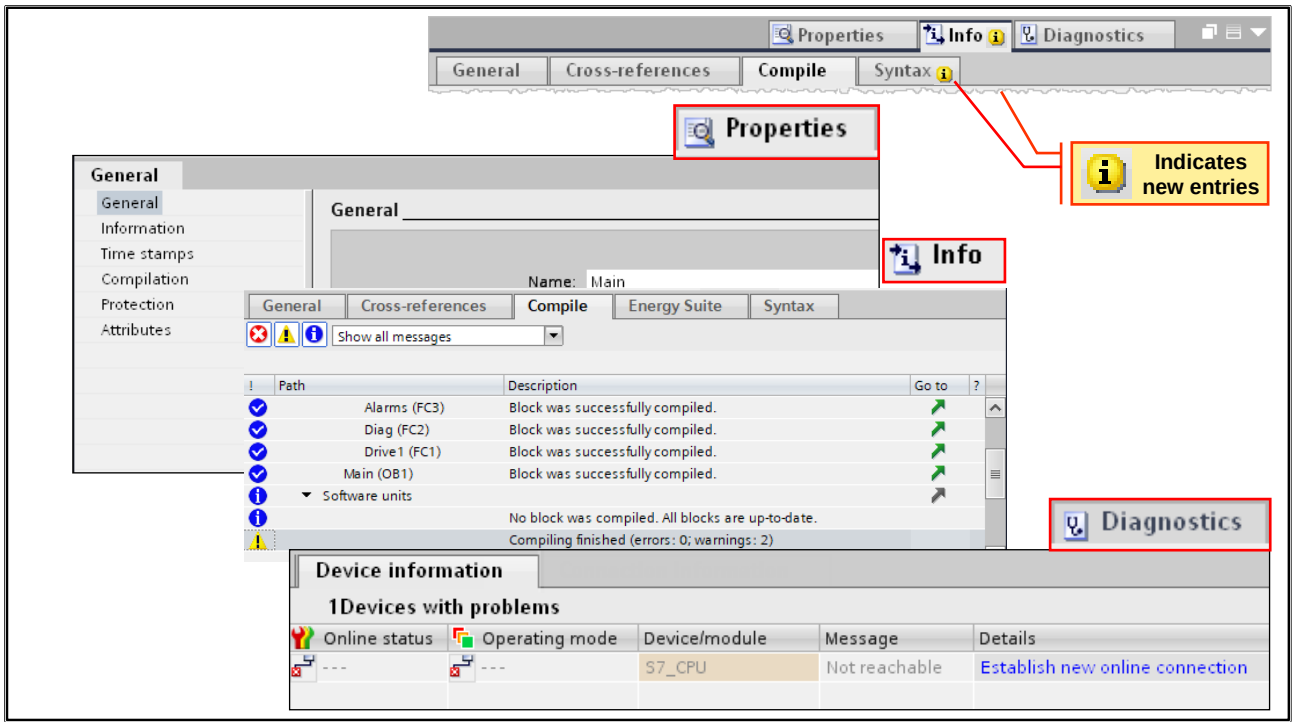
1.6.5. Task cards



Which task cards are available depends on the products installed and the object that was just edited or is open in the work area. If not all task cards can be seen, the task card bar can be scrolled using the cursor buttons on the bottom right.

- **Hardware catalog**
  All available hardware components such as CPUs, modules, etc. can be selected here.
- **Instructions**
  Instructions for block programming;
  Code templates and function list wizard for script programming (VBS or C scripts in WinCC Professional)
- **Online tools**
  When an online connection exists, diagnostic and online information is available, such as the current cycle time of the CPU and the memory space of the load and work memory of the CPU. The CPU can also be switched to STOP and RUN modes.
- **Tools**
  Configurable screen objects (graphics, display and control objects) in different panes (basic objects, elements, controls, optional user controls, graphics)
- **Animations**
  Templates for dynamization of screen objects in different binding for dynamization)
- **Layout**
  Aid for adapting the display during screen design when configuring HMI devices (zoom, layer assignment, grid alignment, objects out of range)
- **Tasks**
  Classic editor functions are available here, such as finding and replacing tags, instructions, etc.
- **Libraries**
  Management of the local project library and global libraries

1.6.6. Inspector window



Additional information on a selected object or on executed actions are displayed in the Inspector window. The Inspector window consists of the following tabs:
→ can be selected by clicking the tabs

This marking in the tab signals new entries.

If errors are indicated, it is possible to jump to the spot of the error or to the appropriate editor by double-clicking the error information.

"Properties" area

This tab displays the properties of the object selected in the work area. Editable properties can be changed here.

"Info" area

This is the output area of the engineering framework. This tab displays additional information on the selected object. Alarms for executed actions, such as compiling and downloading blocks to the CPU, are also output.

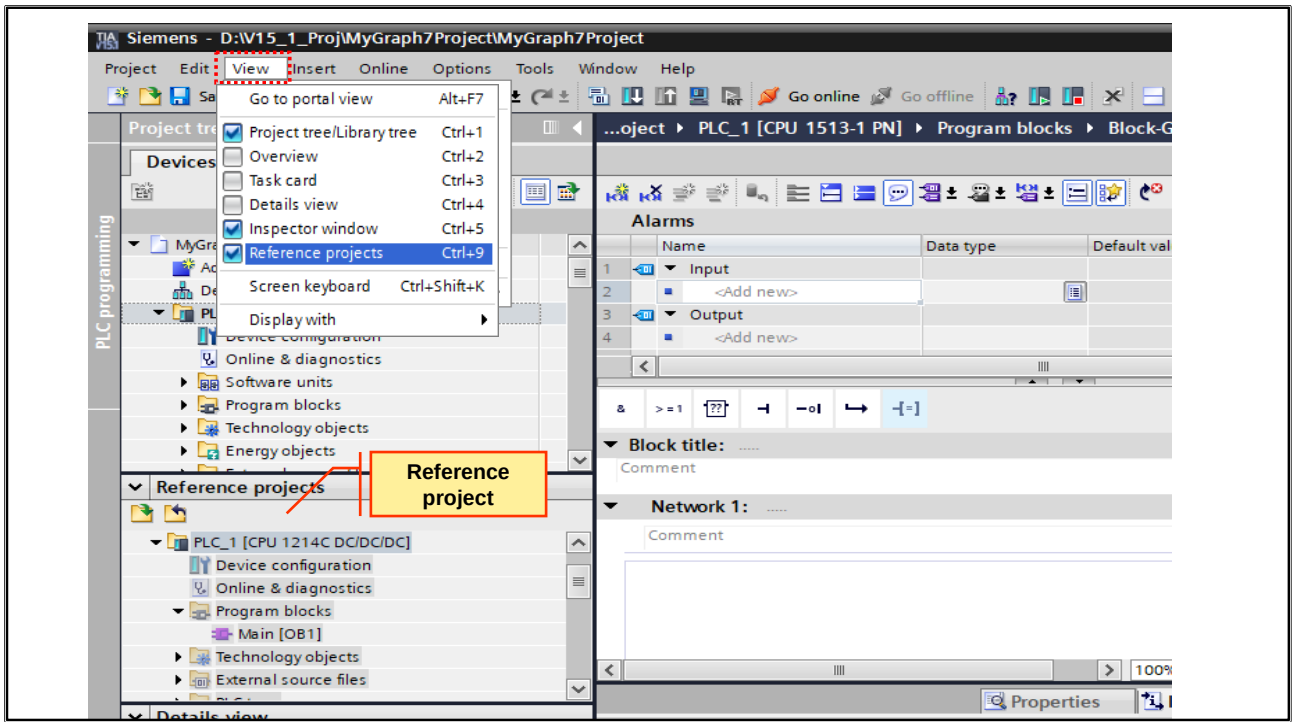
"General" tab → General status information
isplay of the current point of use of the selected object
"Compile" tab → Status output of the compilation
"Syntax" tab → Status output for invalid programming commands

"Diagnostics" area

This tab displays information on system diagnostics and configured alarm events.

"Alarm display" tab → Display of currently pending CPU alarms (e.g. Alarm_S)
"Device information" tab
"Connection information" tab → for components that are currently connected online

1.6.7. Reference project

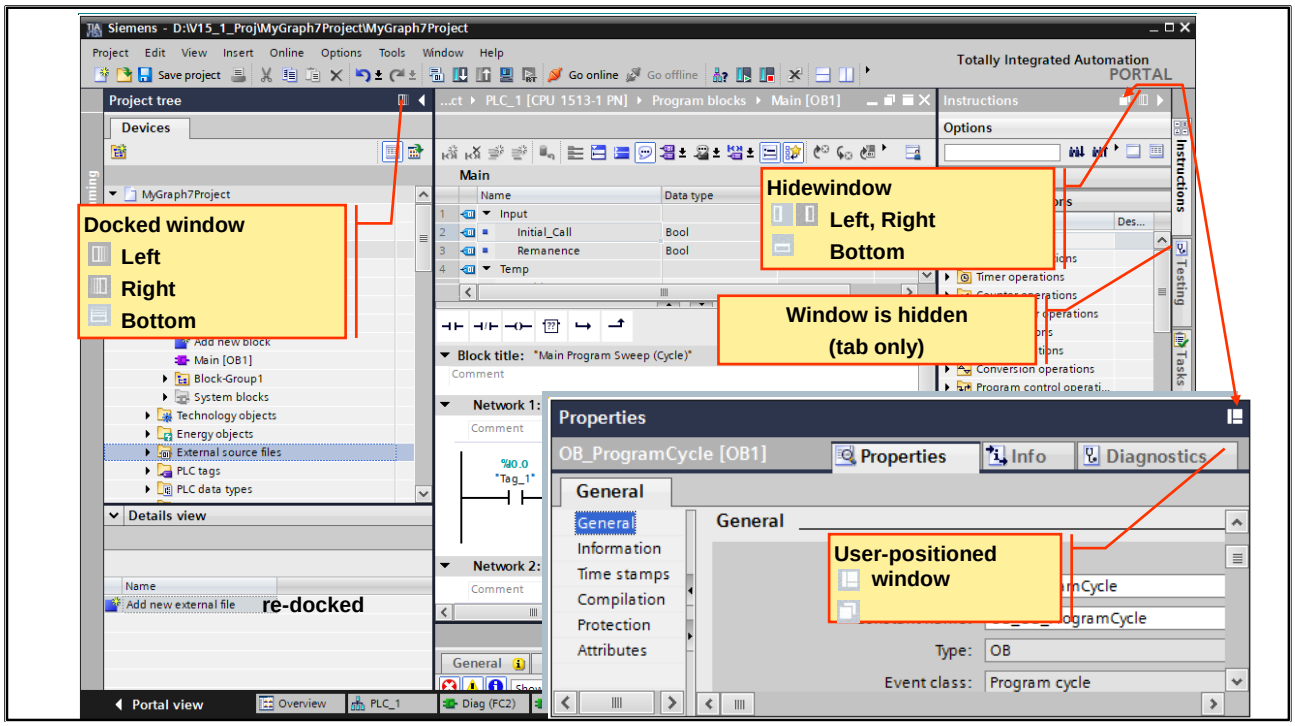


Reference project

In addition to the user project, a reference project can be opened with write protection using the "View" menu (see figure, all project elements have a gray background). Some elements (e.g. program blocks, see figure) can be opened in the editor but not changed.

As in the case of libraries, all project elements can be copied from the reference project to the user project.

1.6.8. Window arrangement in Project view



The position and behavior of windows can be customized. Rarely needed windows can be hidden to increase the surface of the work area.

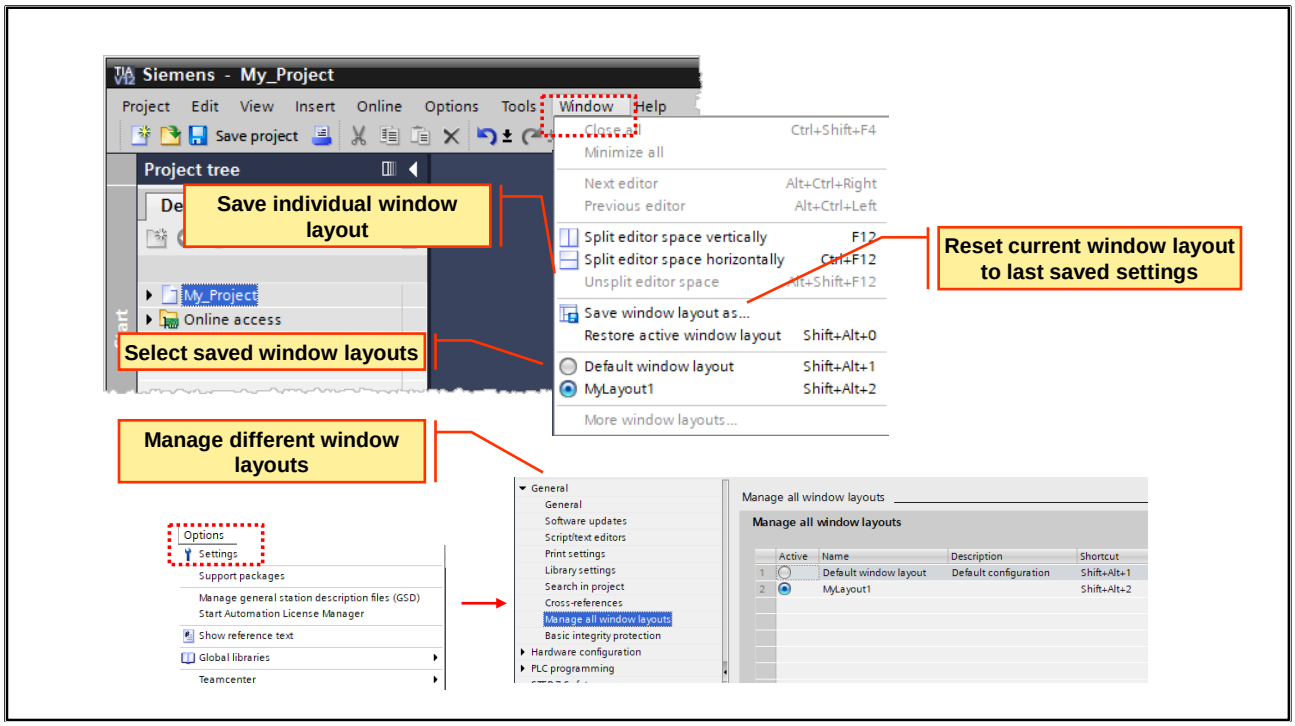
The current configuration of the user interface of the engineering framework is saved in the Windows user profile. When the project is saved, the positions and the behavior of windows are automatically saved as well.

Options for window arrangement

- Docked window
  - Fixed location and size on the user interface
  - Can be positioned on the left of, on the right of or below the work area
  - Always open, reduces the work area
- Hideable window
  - Hidden at the edge of the user interface
  - Can be positioned on the left, on the right or below, covers the work area when opened
  - Default condition = Window closed and tab displayed at edge of user interface
  - Open the window by clicking the tab
  - Automatically closed at the next click outside the window area
- User-positioned window **!! useful when using a second monitor !!**
  - Can be positioned by the user anywhere within the user interface
  - Permanently covers the user interface area underneath

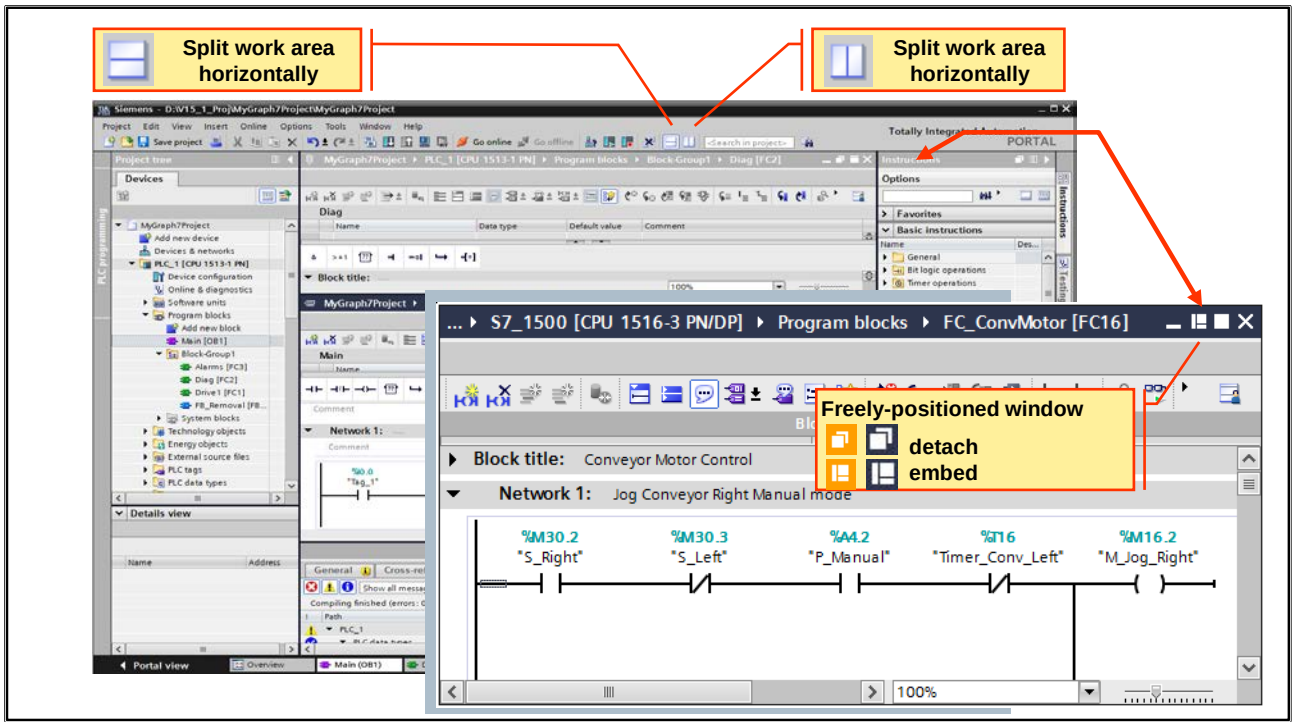
Click the functions in the window header to change between the following modes: "*User* positioned", "*Docked*" and "*Hidden*".
Hidden windows can be opened by clicking their tab and closed again by clicking outside the window area.

1.6.9. Saving/managing/using window layouts



The various window arrangements of the user interface can be saved and restored.

1.6.10. Window layout of work area



The windows of the work area can be arranged as follows:

Maximize the work area (full screen)
(color depending on online/offline view)

Replace a window or take a window out of the work area
(color depending on online/offline view)

Re-embed a window in the work area
(color depending on online/offline view)

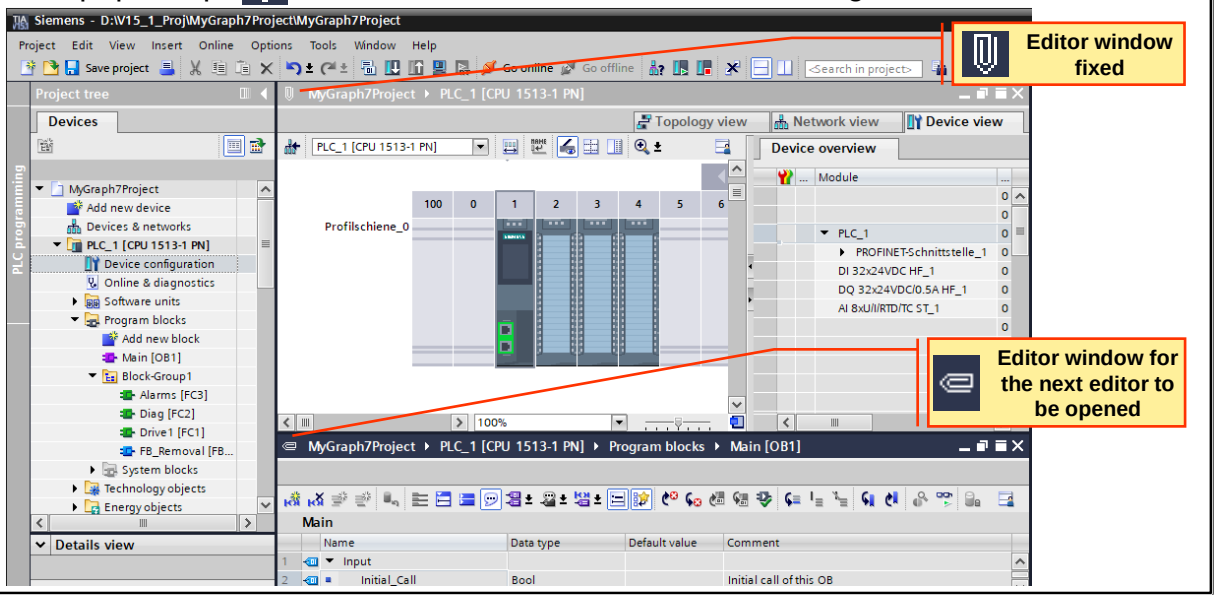
Split work area horizontally into two windows

Split work area vertically into two windows

1.6.11. **Keep editor window in foreground** (with split work area)

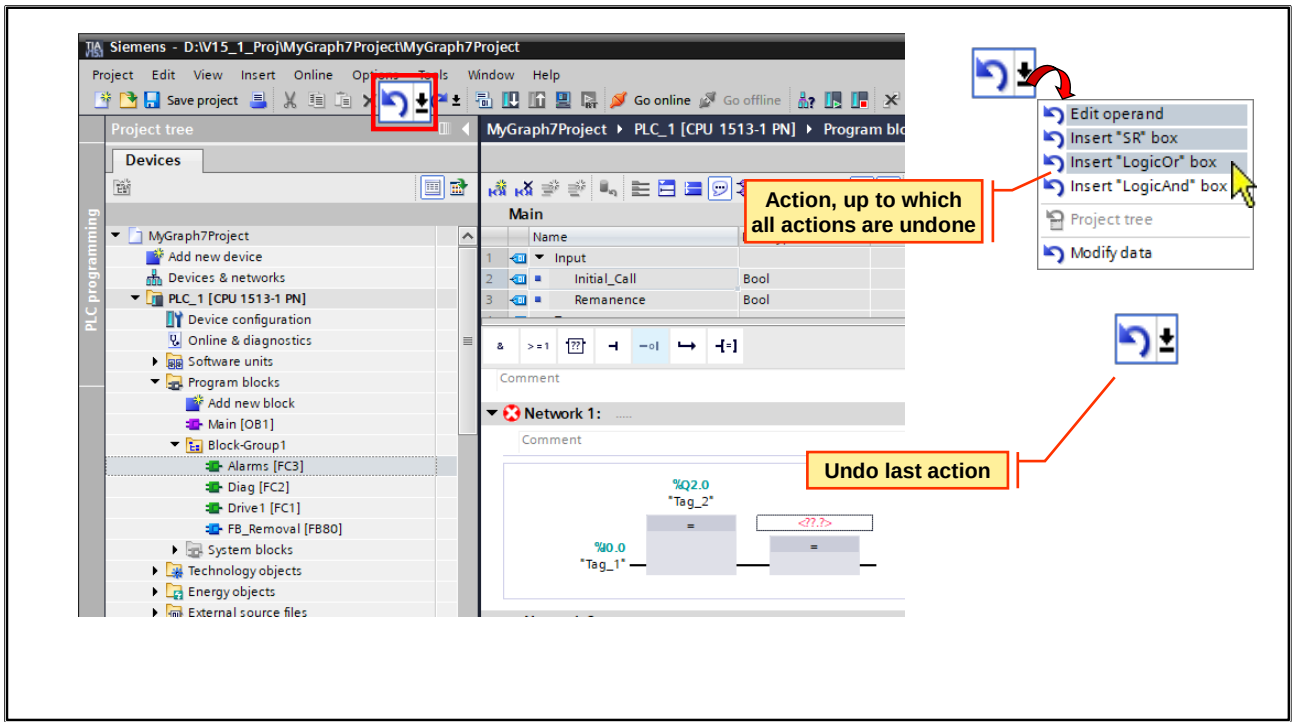
Split work area

→ "paper-clip" ⇒ Editor window "attached" in the foreground



If a split work area is used, one of the two work areas can be fixed by clicking the "paper clip" so that if another editor is opened, the fixed window always remains in the foreground and the newly opened window is always the second visible editor.

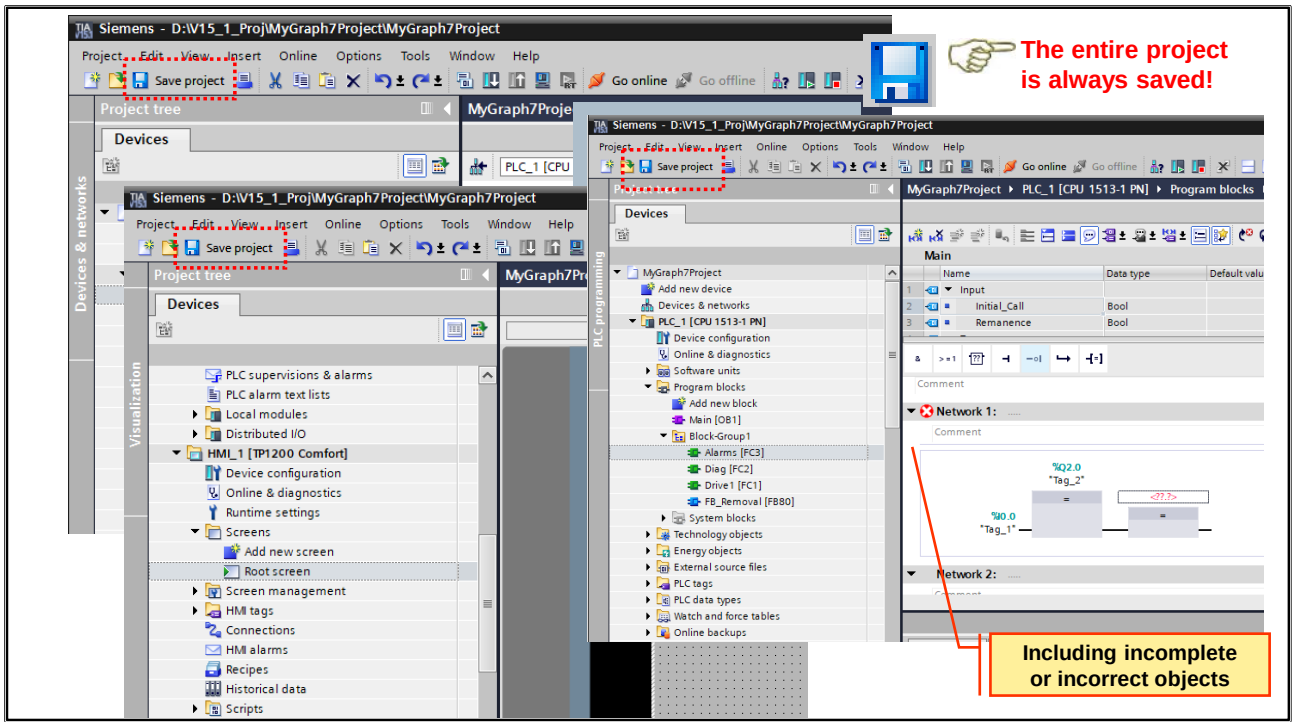
1.6.12. Undo and Redo



Undo concept of the TIA Portal

The drop-down menu shows the user the editor in which the "Undo" function will be executed. Closed editors will be opened automatically for this. Due to the fact that only all actions up to the selected action will be undone, the project consistency is ensured.

1.6.13. Saving a project

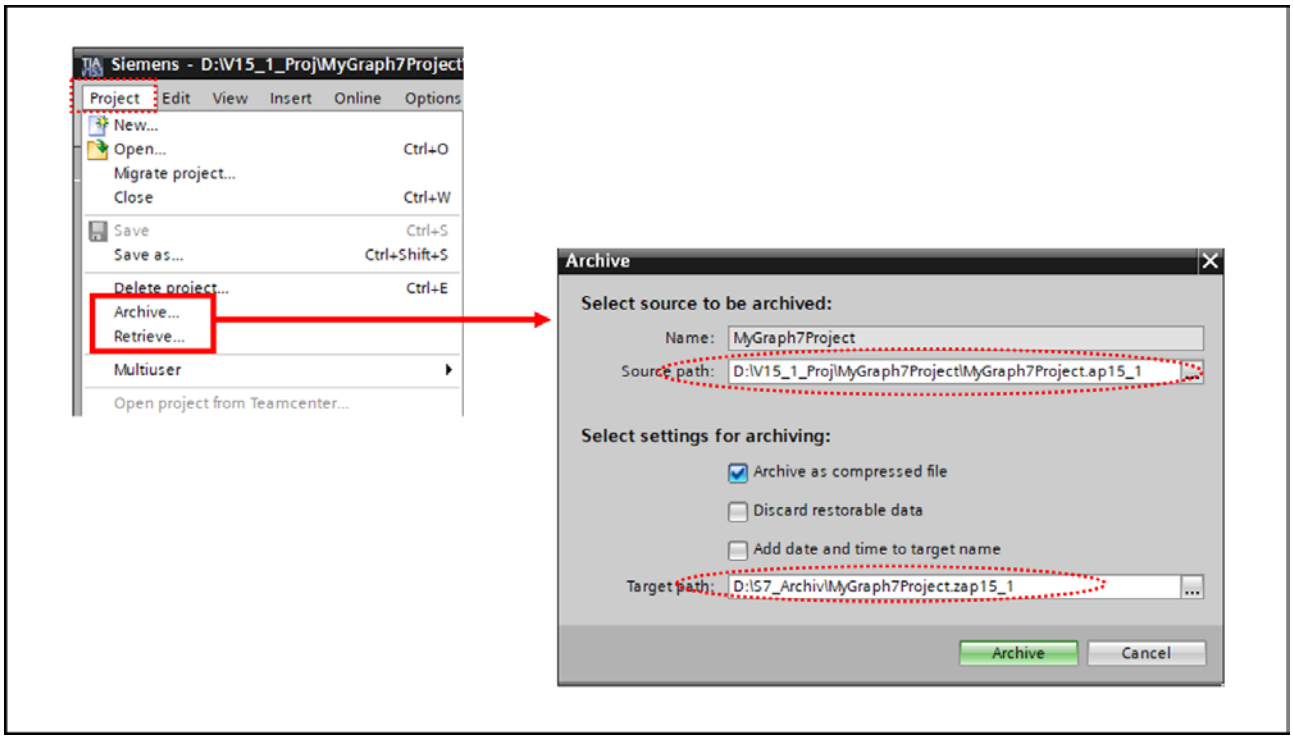

Siemens - D:\V15_1_Proj\MyGraph7Project\MyGraph7Project
Save project
The entire project is always saved!
Including incomplete or incorrect objects

Saving

Regardless of which object is open in the work area, when the Save icon is pressed, the **complete project is always saved** in its current state, even if some objects of the project are still incomplete or have errors (e.g. blocks with syntax errors or symbols that have no absolute operand assigned in the global symbol list).

If the project is closed without saving, all changes made during the session and newly created objects are discarded.

1.6.14. Archiving and retrieving a project



Archiving

You can reduce the required storage space of the currently open project by minimizing the project files, i.e. reducing them to their essential components.

The following can be chosen when archiving projects:

- "TIA Portal project archives"
  The project will be minimized (all files are reduced to their essential components) and then stored in compressed form in a project archive (file with extension *zap16*).
- TIA Portal project minimized
  The project will be minimized (all files are reduced to their essential components) and then stored as a new project in a new project directory.

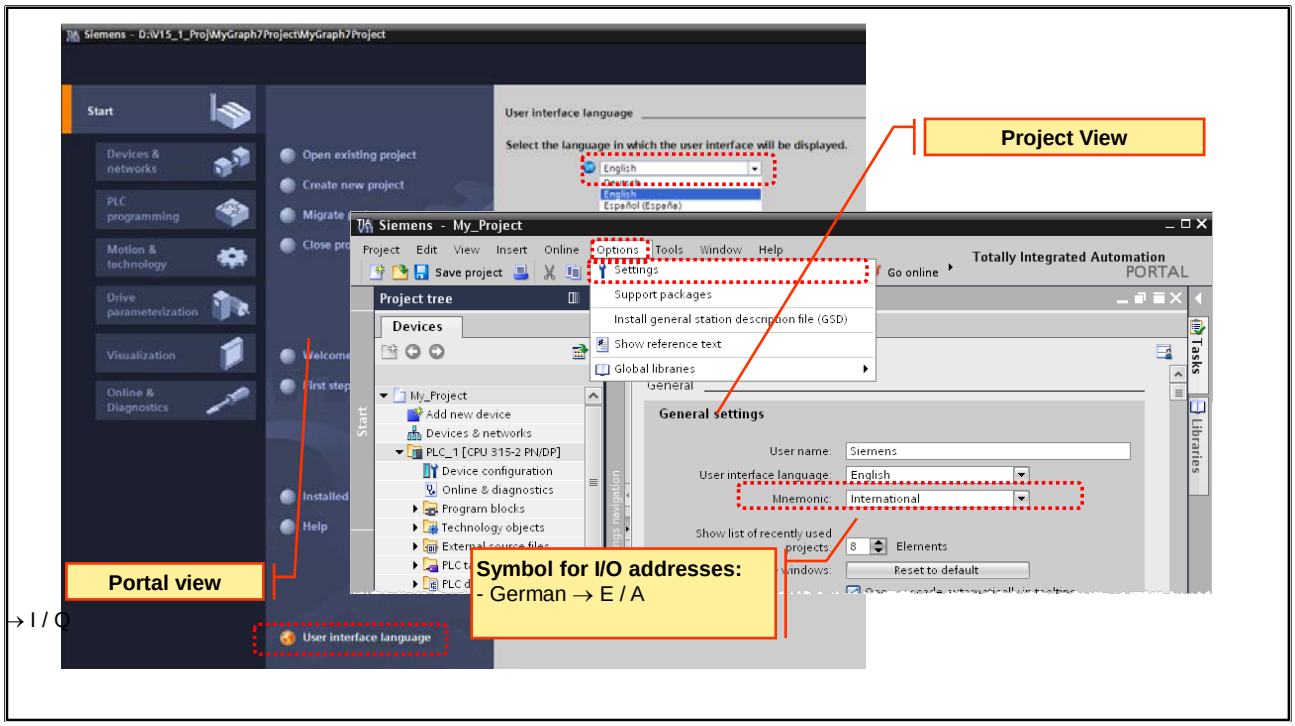
Note

The most recently saved status of the project is archived. If the most recent changes to the project are to be included in the archived project, the project must be saved before archiving.

Retrieving

zap16) can be retrieved and unpacked. Only projects archived with extension

1.6.15. TIA Portal settings: User interface language

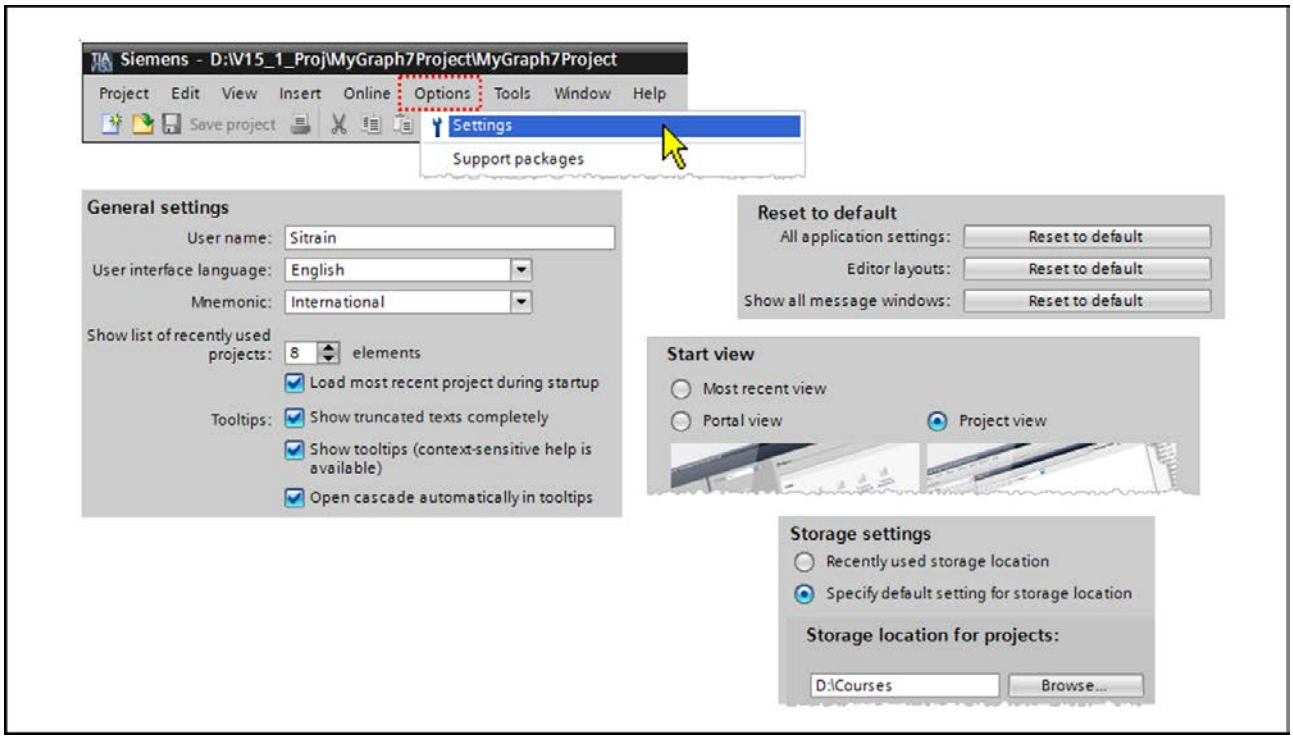


Available user interface languages

The user interface language of the TIA Portal can be changed during operation. The following languages are available:

- German
- English
- French
- Spanish
- Italian
- Russian
- Korean
- Japanese
- Chinese (Simplified)

1.6.16. TIA Portal settings: Language, storage location, layout



Language

The user interface language of the TIA Portal can be changed at any time without having to restart it. The TIA Portal always starts in the same language in which it was last used.

Storage settings

Storage location for projects:
Storage location for newly created projects and their project libraries

Storage location for libraries:
Storage location for global libraries

Layout

When the layout is reset, the original window arrangement of the TIA Portal is restored.

Table of contents 2

2. S7-1500 Training Device 2-2

2.1. Setup of a training station with S7-1500 2-3
2.1.1. Setup and connection of the conveyor model to the S7-1500 2-4
2.2. Positioning of the modular S7 controllers 2-5
2.3. SIMATIC S7-1500: Modular controller for medium to upper performance range 2-6
2.3.1. SIMATIC S7-1500: Modules 2-7
2.3.2. SIMATIC S7-1500: CPU display → Menu and colors 2-9
2.3.3. SIMATIC S7-1200/1500: Memory card(s) 2-10
2.4. Distributed I/O systems 2-11
2.5. Modules of the S7-1500 training device 2-12
2.6. Modules of the ET 200SP training device 2-13
2.7. Networking and IP addresses of the modules 2-14
2.8. Online tools, configuring and parameter assignment of the hardware 2-15
2.9. Online access: Accessible devices in the Portal view 2-16
2.9.1. CPU online access: Reading out the diagnostics buffer 2-17
2.9.2. CPU online access: CPU-RUN/STOP, Memory Reset (MRES) 2-18
2.9.3. Online access to the CPU: IP address, name, time, FW update, format memory card 2-19
2.10. Memory reset (MRES) of the CPU via the mode switch 2-20
2.11. Devices & networks: Device view (hardware configuration) 2-21
2.12. Preset configuration: Creating a hardware station 2-22
2.12.1. Inserting/deleting a module 2-23
2.13. Downloading actual configuration to project (1): Inserting an unspecified CPU 2-24
2.13.1. Downloading the actual configuration to the project (2): Determining accessible nodes 2-25
2.13.2. Downloading the actual configuration to the project (3): Reading out the actual configuration 2-26
2.14. Compiling the hardware configuration and downloading it to the CPU 2-27
2.15. CPU properties: Ethernet address 2-28
2.16. Task: Hardware – Commissioning 2-29
2.16.1. Exercise 1: Connect PG and set IP address of the PG 2-30
2.16.2. Exercise 2: Erase SIMATIC memory card of the CPU 2-31
2.16.3. Exercise 3: Reset the CPU via the mode switch 2-32
2.16.4. Exercise 4: Determine CPU firmware version and assign IP address (node initiation) 2-33
2.16.5. Exercise 5: Configure S7-1500 station 2-34
2.16.6. Exercise 6: CPU properties: IP address and subnet mask 2-36
2.16.7. Exercise 7: Compile the hardware configuration and download it to the CPU 2-37

2. S7-1500 Training Device

The participant should ...

... know the setup of the training station

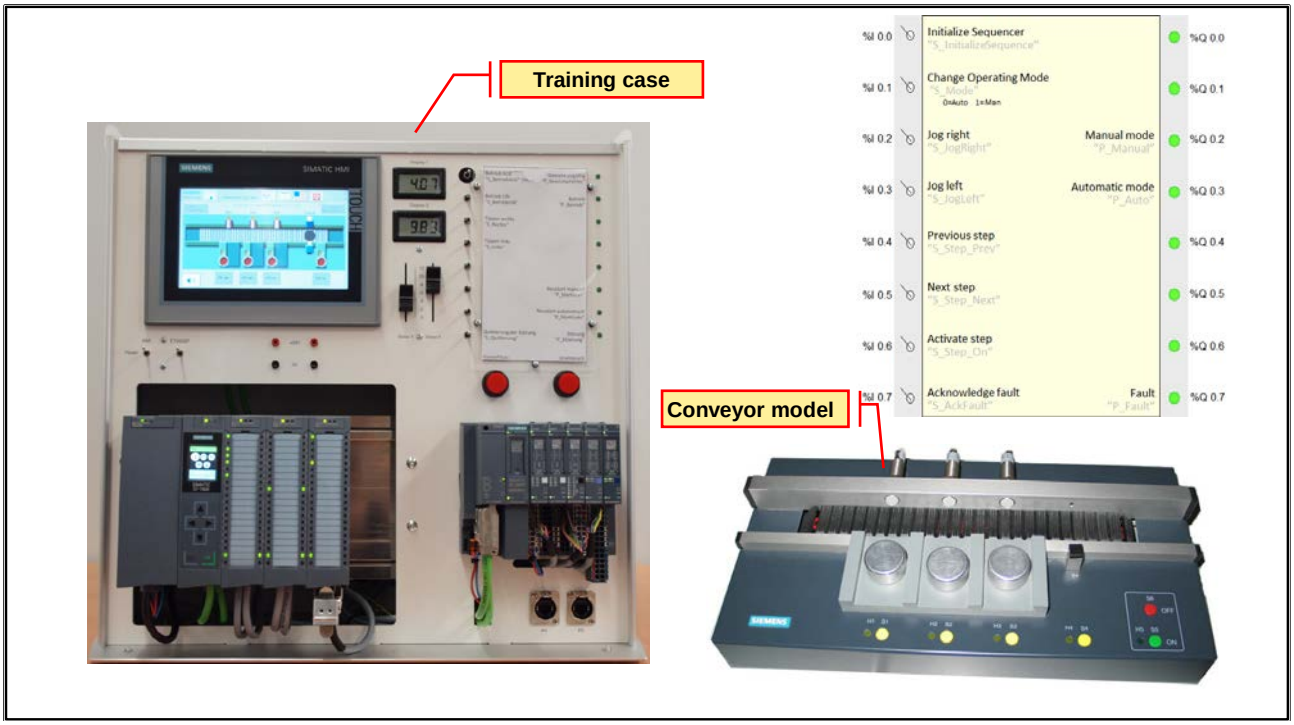
... know the wiring of the training station components

... be able to establish an online connection between the PG and CPU

… be able to use online functions to stop and start the CPU and reset it to factory settings

... be able to create and configure a preset configuration

2.1. Setup of a training station with S7-1500

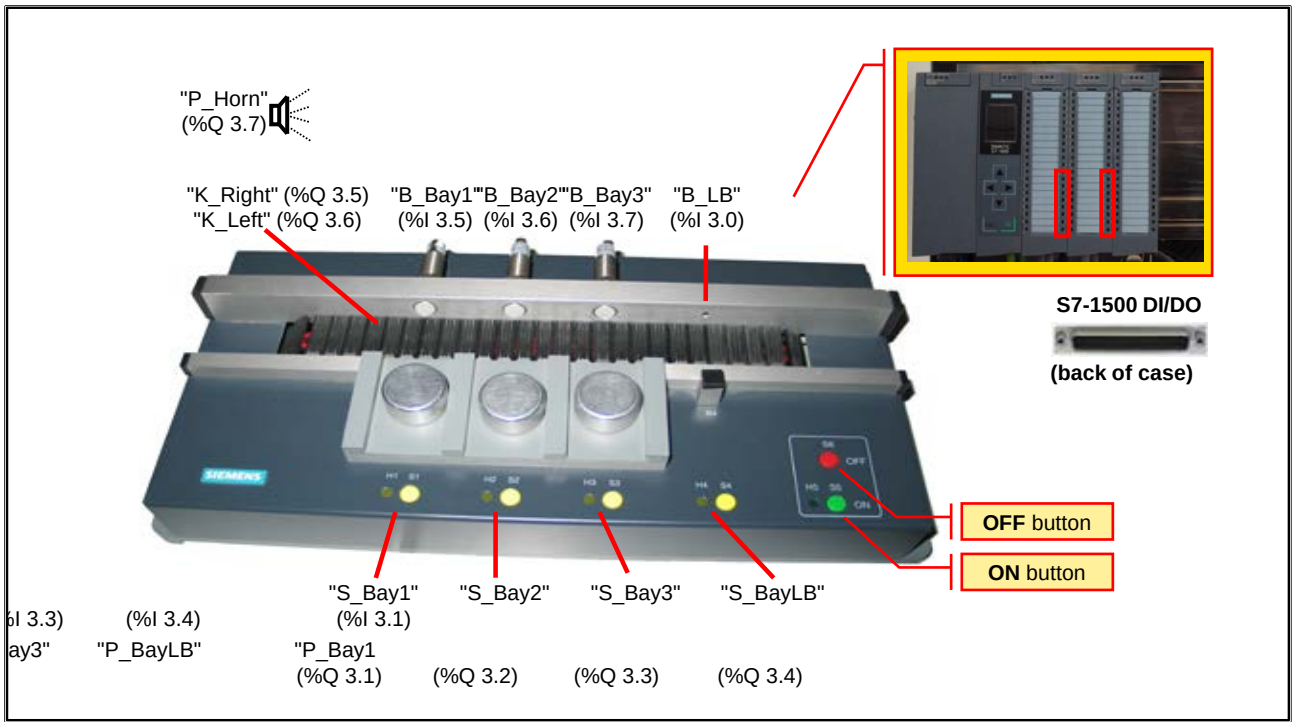


Setup of a training station with S7-1500

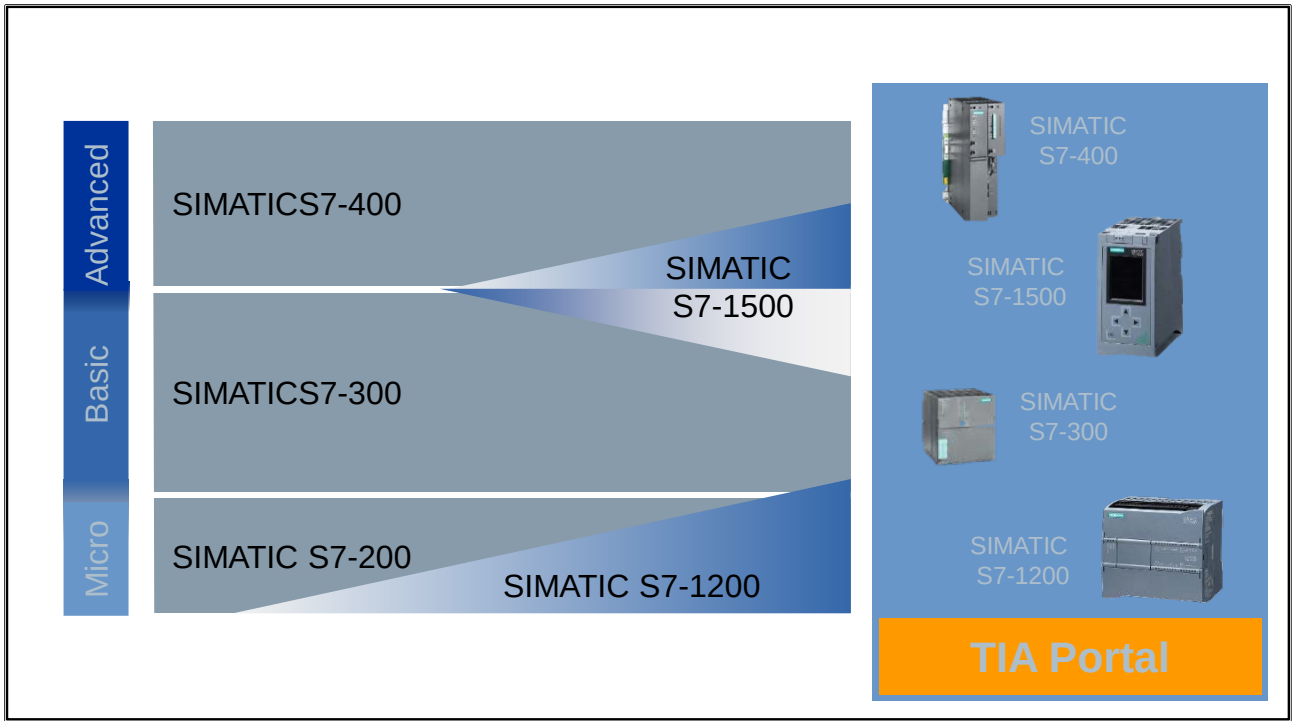
The training station for this course includes the following components:

- SIMATIC Field PG
- Training case with S7-1500, ET 200SP, touch panel, control elements (switches/pushbuttons and sliders) and display elements (lamps and voltage displays)
- Conveyor model

2.1.1. Setup and connection of the conveyor model to the S7-1500



2.2. Positioning of the modular S7 controllers



SIMATIC S7

The SIMATIC S7 programmable logic controllers can be classified according to performance range: a micro PLC (S7-1200) and a medium/upper performance range PLC (S7-1500).

The existing low/medium performance range PLC (S7-300) and the medium/upper performance range PLC (S7-400) will be covered by one S7-1500 system in the future.

2.3. SIMATIC S7-1500: Modular controller for medium to upper performance range

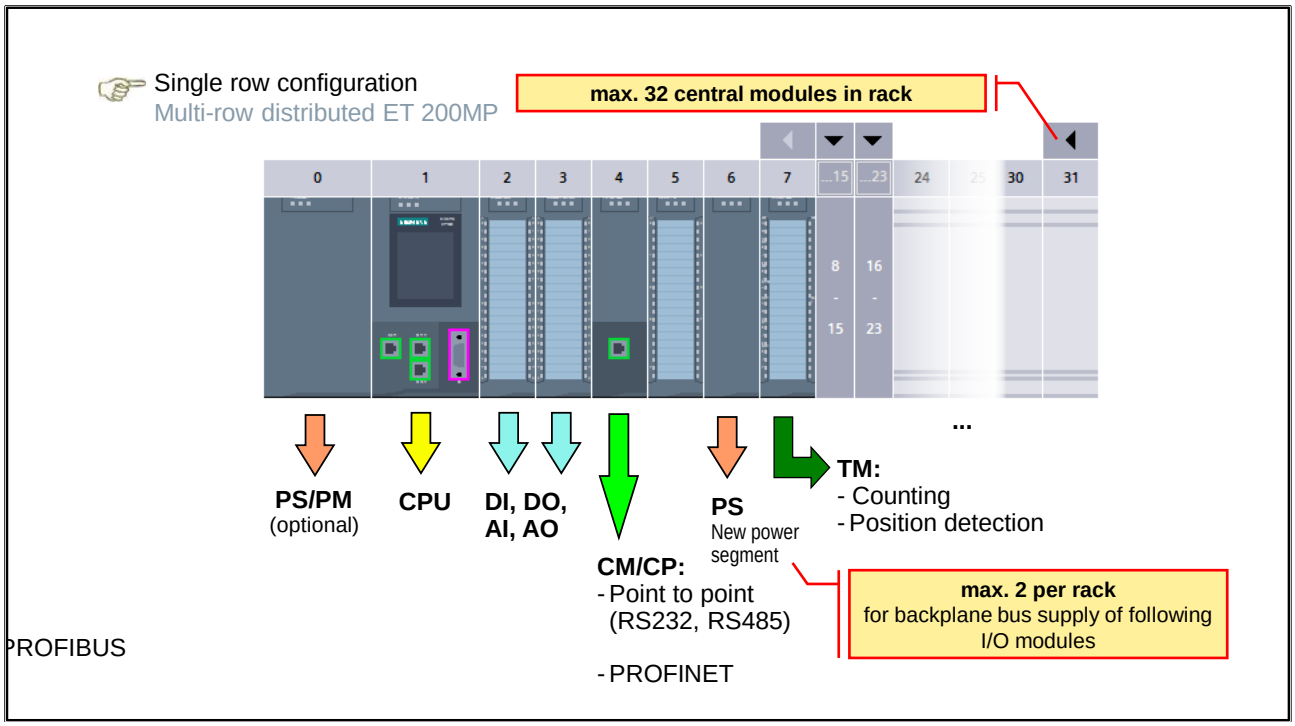
Features

- Modular control system for medium to upper performance range
- Graduated range of CPUs
- Display for basic CPU settings (system time, interfaces, etc.) and for calling up diagnostic and status information (diagnostic buffer, alarm display, CPU status, etc.)
- Comprehensive range of modules
- High-performance I/O bus for efficient process interfacing via central I/O
- Can be expanded to up to 32 modules in one tier.

  Currently, only single-tier configuration is possible. Multi-tier configuration via ET 200MP distributed I/O system.

- Can be networked with PROFIBUS or PROFINET
- Slot rules
  - Left of the CPU:
    1x power supply (PM or PS)
  - Right of the CPU:
    Signal modules (digital, analog), technology modules, communication modules and other power supplies
- No slot rules for modules that are mounted to the right of the CPU

2.3.1. SIMATIC S7-1500: Modules



Slot rules

- 1x PS/PM Slot 0
- 1x CPU on Slot 1
- Any module starting from slot 2

Signal modules

- Digital input modules: 24V=, 230V~
- Digital output modules: 24V=, 230V~
- Analog input modules: Voltage, current, resistance, thermocouple
- Analog output modules: Voltage, current

Communication modules (CP – Communication processor, CM – Communication module)

- Point-to-point connection
- PROFIBUS
- PROFINET

CPs and CMs are both communication modules.
CPs generally have more functionality than CMs (e.g. own web server, firewall and the like).

Technology modules (TM)

- Counters
- Position detection

Power supply

I/O modules in the central rack of the S7-1500 require a system power supply via the backplane bus (communication connection to the CPU) and a load power supply (input and output circuits for sensors and actuators).

- **PM – Power Module → Load power supply**
  supplies modules with 24 VDC for input and output circuits as well as sensors and actuators

  If the CPU is supplied with 24 V via a load power supply (PM), this supplies the system power supply of 12 W for the first inserted I/O modules.

- **PS – Power Supply → System power supply**
  supplies S7-1500 modules in the central rack via the backplane bus

  Each CPU provides a system power supply of 12 W for the first inserted I/O modules. Depending on the I/O modules used, however, additional power segments must be configured if necessary.

  A system power supply (PS) can supply the CPU as well as the load circuit for 24 VDC modules.

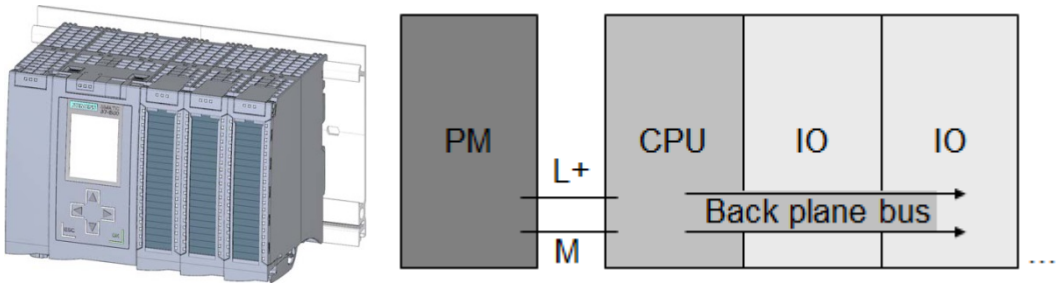
Power supply and power segments of the I/O modules

The formation of power segments is necessary for larger assemblies or assemblies with I/O modules in the central rack with high power demand (generally when using CP, CM, TM).

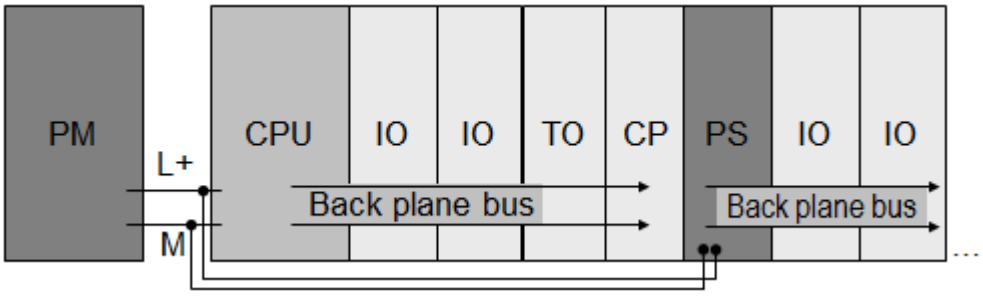
A maximum of 3 power segments per rack can be formed (1x CPU segment plus 2 others).

When the configuration variant with additional power segments is used, additional system power supply modules (PS) are used just to the right of the CPU. The CPU continues to control all modules of the rack. Only the system power supply of the I/O modules is subdivided here.

Example of a small S7-1500 configuration



Example of an S7-1500 configuration with a 2nd power segment



Interface connections for expansion rack

A multi-tiered central configuration is not planned. An expansion is possible using the ET 200MP distributed I/O system.

2.3.2. SIMATIC S7-1500: CPU display → Menu and colors

Main menu items and their meaning:

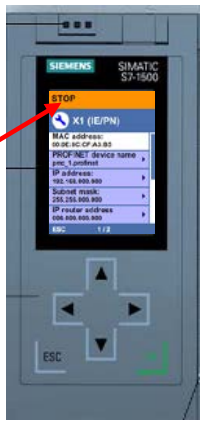
Overview (CPU status: name, type, MLFB, version)

Diagnostics (diagnostic buffer display, alarm display)

Settings (CPU settings: addresses, time, operating mode, CPU reset, protection level, lock display when password set)

Modules (status information of modules in the system)

Display (display settings: brightness, display language, power saving, MLFB, version)



Colors of status information and their meaning:

Green	RUN of CPU, RUN with pending alarm
Yellow	STOP or HALT of CPU
White	Connection setup or connection to CPU lost

The colors in the display vary depending on the current CPU status.

Available display languages are the available user interface languages of STEP 7.

Additional symbols in the status information

Password is configured but not entered

Password is configured and entered

An alarm is present

A force job is active on the CPU

2.3.3. SIMATIC S7-1200/1500: Memory card(s)

① Serial number of SD card
② Product version
③ Order number
④ Card size
⑤ Slider for write protection
(must not be write-protected)

SIMATIC memory card in the S7-1200 / S7-1500:

- Load memory
- Distribution of programs (S7-1200 only)
- Firmware update
- Documentation
- Memory card binding
- Unlinked DBs
- Logging of data
- Module replacement without PG

Written with:

- Commercially available SD card reader
- Field PG

Memory card binding – Copy protection

It is possible to bind the ability to execute the program to the serial number of the card.

Load memory

- S7-1500
  This has no integrated load memory and requires a card to be inserted for this.
- S7-1200
  This has integrated load memory. An inserted memory card can replace the integrated load memory (increase the memory) or the card can be used for program updates (distribution of programs).

Distribution of programs ←S7-1200 only

Use as a transfer card (card mode = "Transfer") is only supported by the S7-1200. Here, a program can be downloaded to the CPU without a PG by inserting a card.

Archiving of data ← S7-1500 only

It is possible to archive process values on the card.

The use of this functionality influences the service life of the memory card

2.4. Distributed I/O systems

Distributed installation

with modules of the central installation
- Front connector
- 1-wire system
- Optimized width

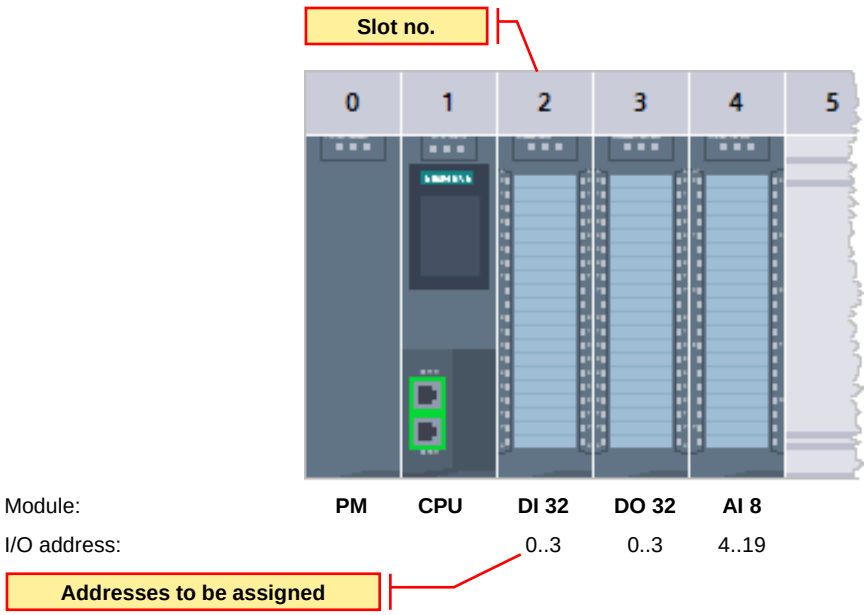
Distributed installation
- Stationary wiring and multi-wire system
- Toolless, fast connection system
- Optimized size

One distributed I/O for various market segments.

Distributed installation
- 24 V DC and 230 V AC power supply

Functionality for process automation

2.5. Modules of the S7-1500 training device

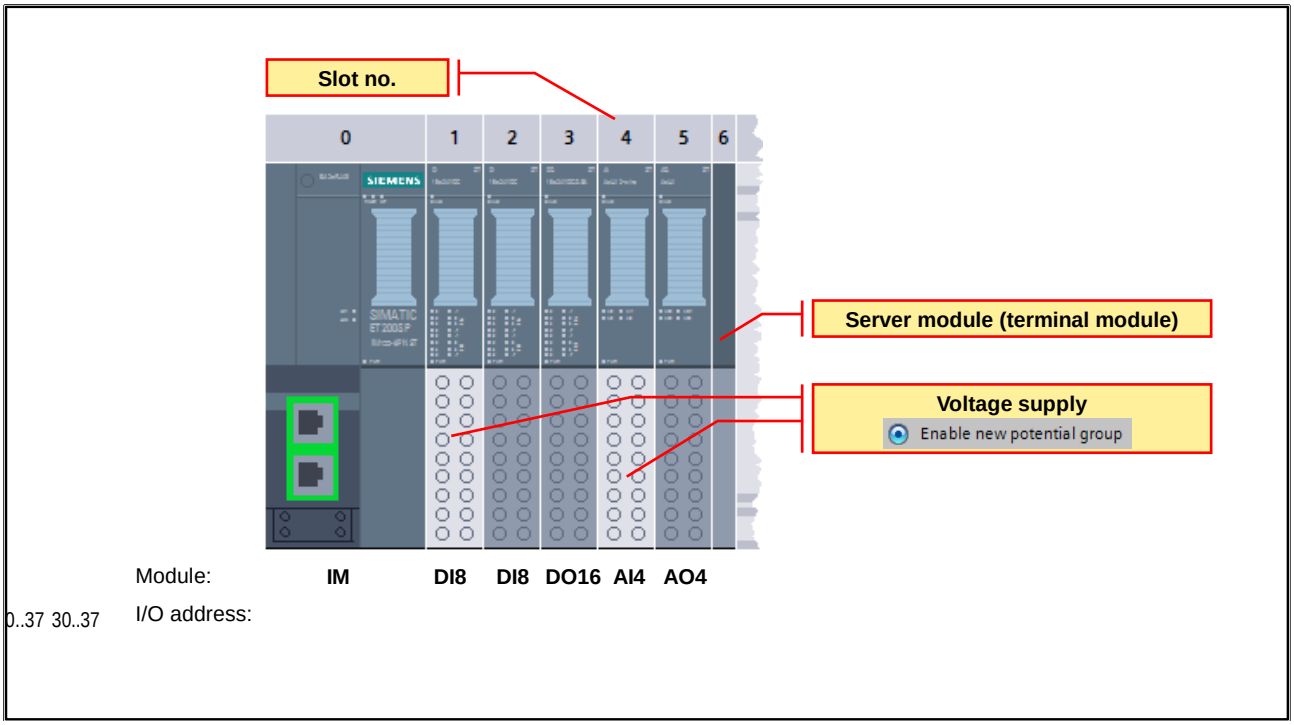


Addresses of the central S7-1500 I/O modules.

Two digital 32-channel modules are available as central modules. These are to begin starting from Address = 0.

Because digital channels are also available on the distributed I/O, the analog module of the central I/O is to begin starting from Address = 10.

2.6. Modules of the ET 200SP training device

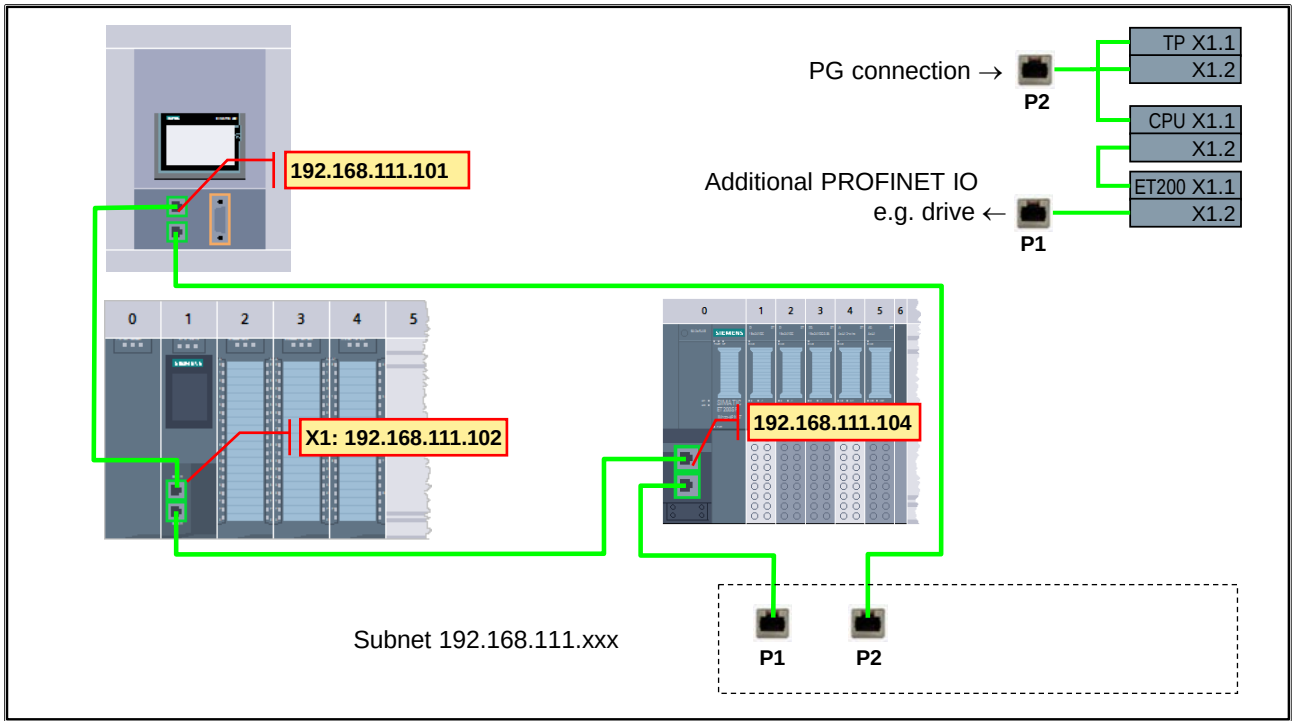


Addresses of the distributed ET 200SP I/O modules

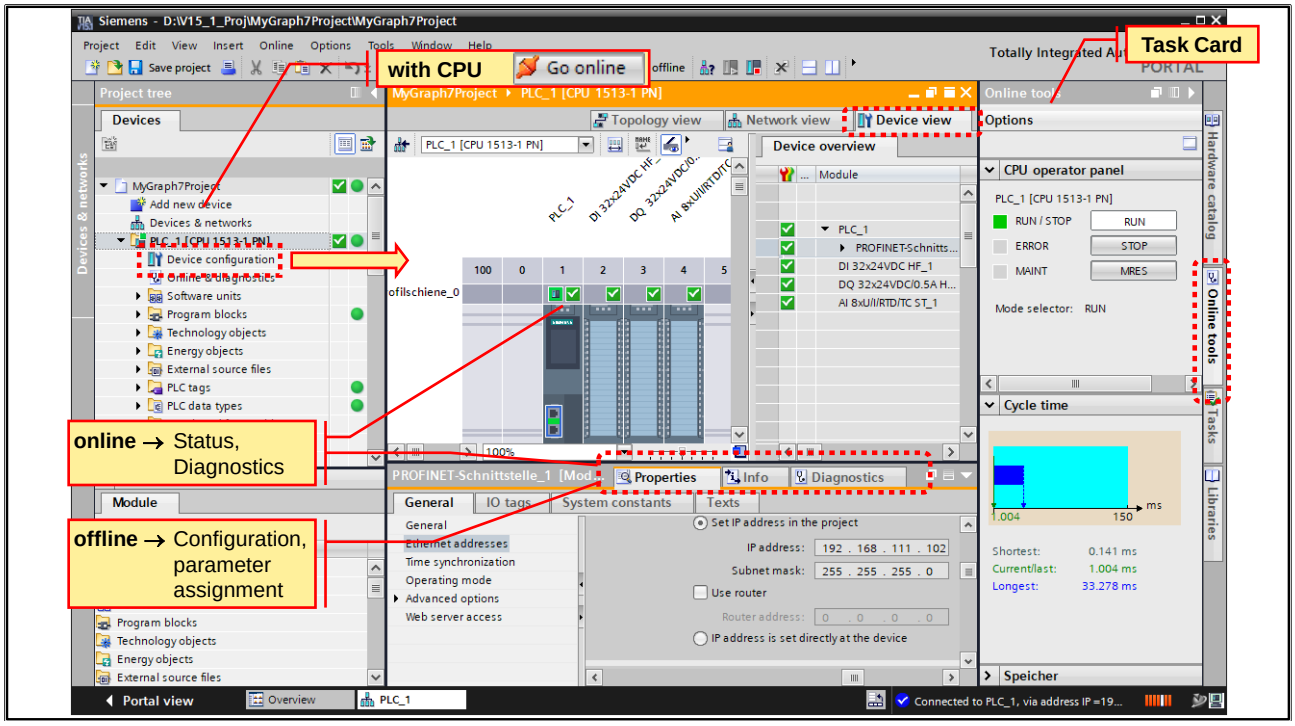
Three digital modules are available as distributed I/O. These are to be connected in the address space to the central digital I/O starting from Address = 4.

The analog distributed I/O is to begin starting from Address = 30.

2.7. Networking and IP addresses of the modules



2.8. Online tools, configuring and parameter assignment of the hardware



Online tools

If an online connection to the CPU can be established, diagnostic and status information of all modules can be called up.

In the case of CPUs that are accessible online, the operating mode can also be controlled using the "Online Tools" task card and additional status information (cycle time statistics and memory utilization) can be called up.

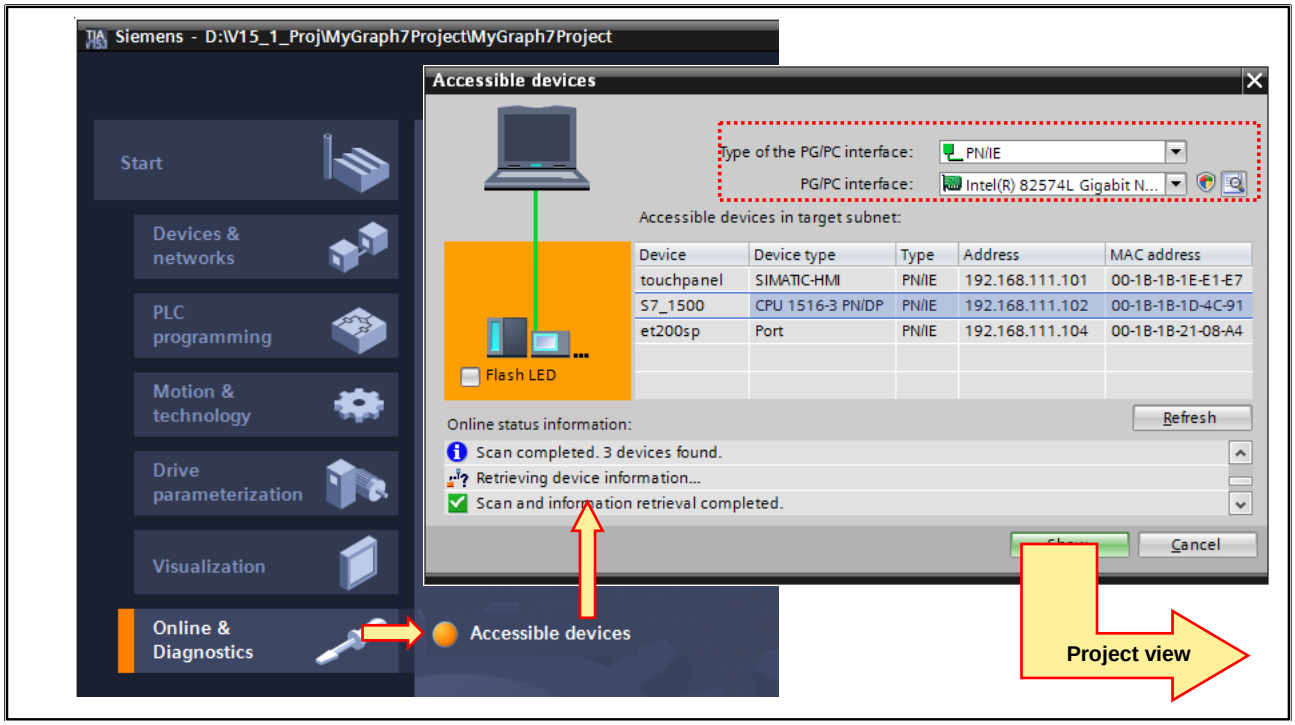
Configuration and parameter assignment of the hardware

Almost all devices and components of an automation solution, such as controllers and touch panels, can be configured. The device parameters and network settings required for commissioning are assigned in the "Devices & networks" editor.

For example, IP addresses are assigned to all components of an Ethernet network. The components then use these IP addresses to communicate with one another during subsequent operation.

Other settings within the automation device include the address ranges of the I/O modules and the cycle monitoring time of the CPU.

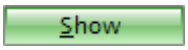
2.9. Online access: Accessible devices in the Portal view



Accessible devices in the Portal view

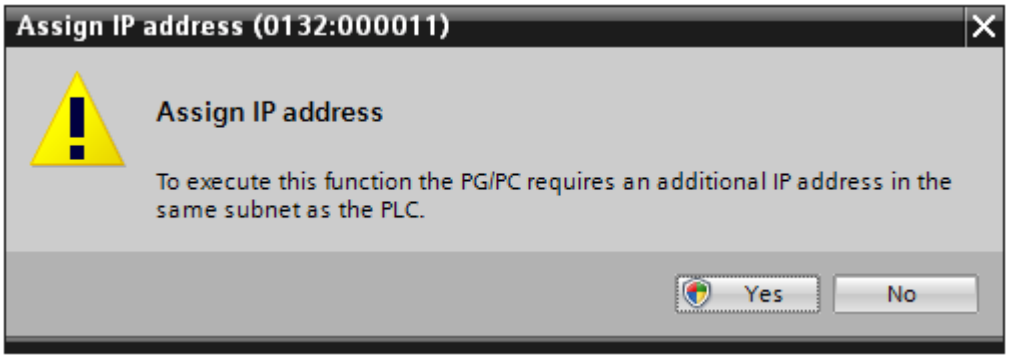
This function enables fast access (e.g. for service purposes) even if no offline project data for the target systems is present on the PG.

In the Portal view, all accessible programmable modules (CPUs, FMs, CPs, HMI devices) are listed even if these are located in other subnets.

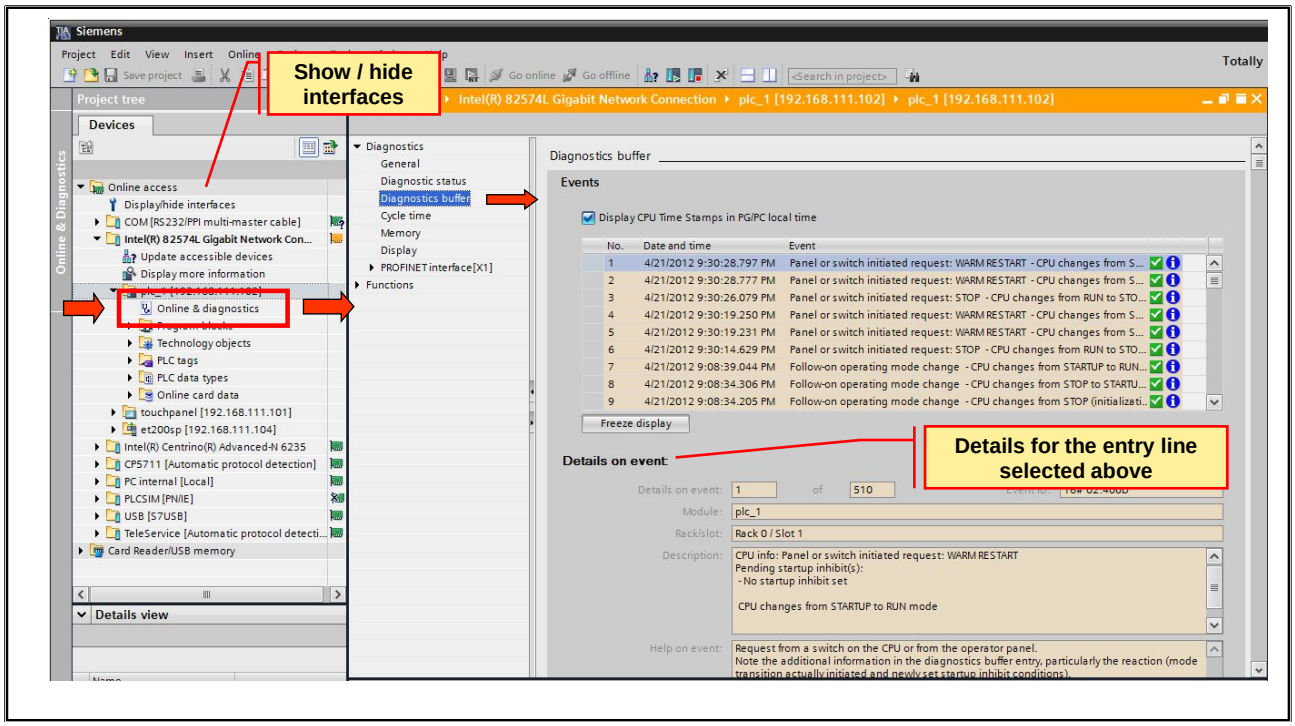
Accessing online functions → button 

As soon as a module that is located in another subnet than the PG is to be accessed online using the "Show" button, a prompt appears in a dialog indicating that an additional IP address is required to be assigned to the PG.

When confirmed, an additional IP address that is located in the same subnet as the CPU address is assigned to the PG. All online functions can then be used.


Assign IP address (0132:000011)
Assign IP address
To execute this function the PG/PC requires an additional IP address in the same subnet as the PLC.
Yes
No

2.9.1. CPU online access: Reading out the diagnostics buffer



Online access to the CPU

If the PG and the target system (e.g. CPU) are in the same subnet, various online & diagnostics functions are available in the "Accessible devices" function

- in the work area of the TIA Portal
- in the "Online tools" task card

Diagnostics buffer

As a circular buffer, the diagnostics buffer is an organized buffered memory area of the CPU. It contains all diagnostic events (error messages, diagnostic alarms, startup information, etc.) of the CPU in the order in which they occurred. The entry at the top is the last event that occurred.

All events can be displayed in plain text and in the order of their occurrence on the programming device.

The size of the diagnostics buffer is CPU-dependent. Likewise, the whole diagnostics buffer is not buffered at a Power OFF (only a portion is retentive).

- Number of entries, 1000..3200

-proof

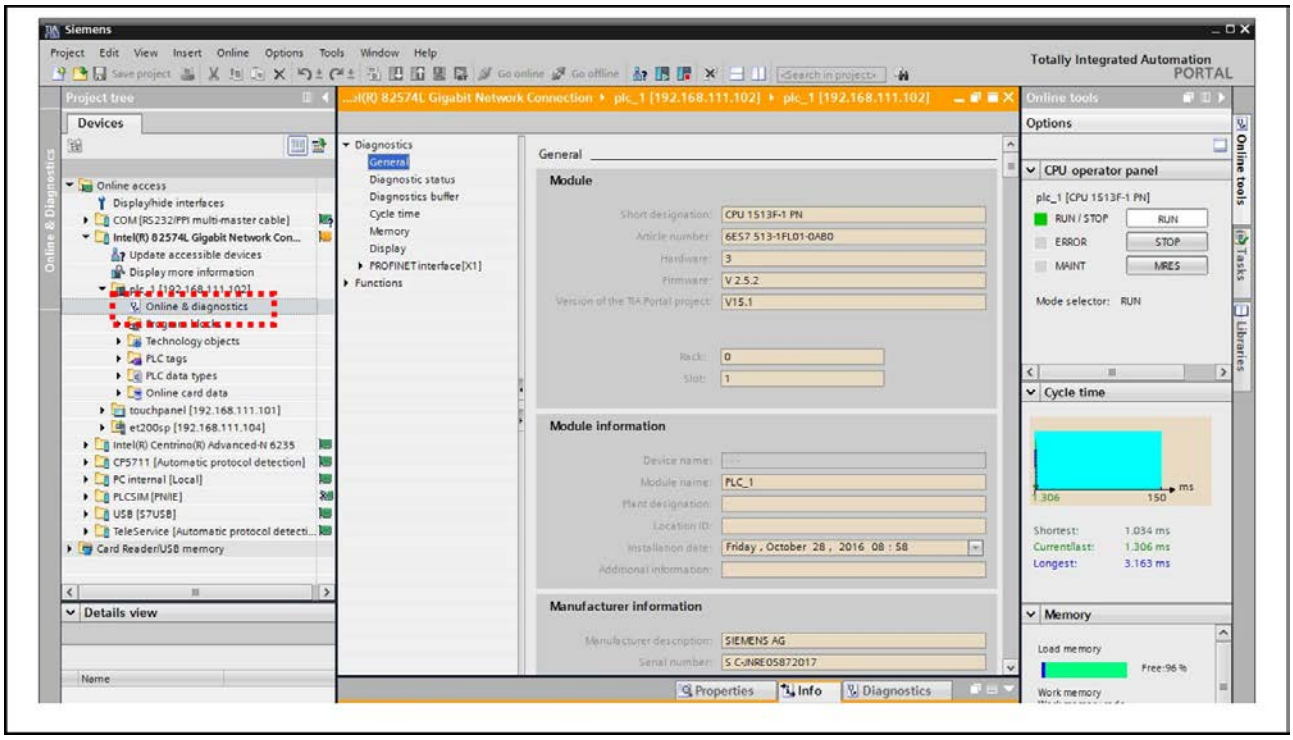
- of which 250...500 are power failure

Details on event

The "Details on event" field provides additional information on the selected event:

- Designation of the event and event number
- Additional event-dependent information, such as the address of the instruction that caused the event

2.9.2. CPU online access: CPU-RUN/STOP, Memory Reset (MRES)



CPU operator panel: Operating mode selection

The CPU operating mode can be switched.

- RUN → STOP:
  At a switch from RUN to STOP, the CPU ends the running user program.
- STOP → RUN:
  At a switch from STOP to RUN, the CPU performs a restart.

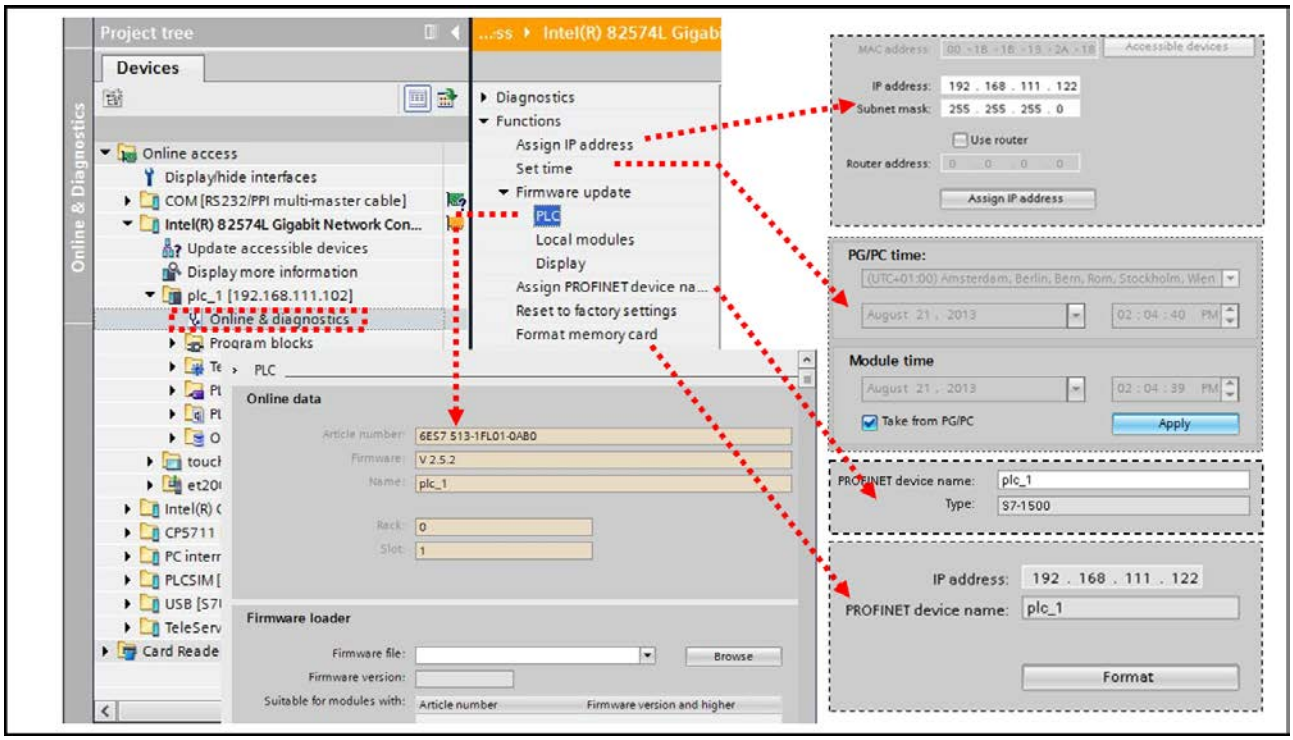
Cycle time:

"Shortest", "Current/last" und "Longest" are the cycle times since the last CPU restart

At a memory reset (MRES) of the CPU, a memory reset is performed:

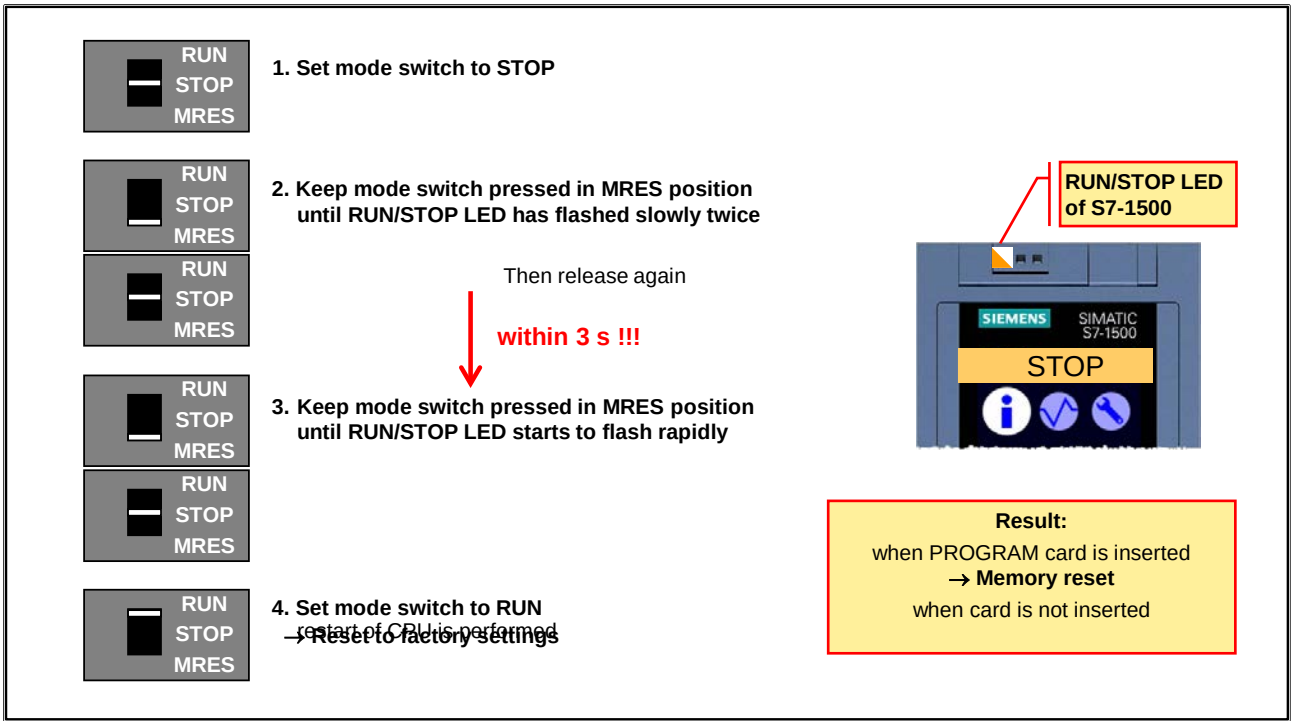
- All user data (including retentive data) is deleted (erasing of work memory)
  (process image, bit memory, counters, timers, program/data blocks)
- Retained: Parameter assignment of the X1 interface, retentive part of the diagnostics buffer, time

2.9.3. Online access to the CPU: IP address, name, time, FW update, format memory card



- **Set time**
  Each S7-CPU has a real-time clock that can be set here.
- **Assign IP address**
  As long as an IP address is not already specified via a previously downloaded hardware configuration, it can be assigned or changed (this function is also available if the PG/PC and the CPU are not assigned to the same subnet).
- **Reset to factory settings**
  In contrast to "Memory reset", all memory areas of the CPU (work, load and retentive memory, diagnostics buffer and time) are erased here. Optionally (see dialog in the screen), the IP address can also be deleted so that the CPU is then only available via a MAC address.
- **Format memory card**
  The memory card of the CPU can also be erased in the CPU using this online function. The CPU then has only its IP address. All other data (including the device configuration) is deleted.
  The card cannot be erased in the card reader via the project tree. Device configuration and blocks have a gray background, i.e. are write-protected (only status information or open with double-click).
- **Assign name**
  With PROFINET each device must be assigned a unique device name that is retentively stored in the device. The device names will be used to identify a distributed I/O module (PROFINET IO) and will enable a module replacement without a PG/PC.
- **Firmware update**
  The firmware version of the CPU can be updated here. The current firmware version of the CPU is displayed in the module data under "Diagnostics -> General".

2.10. Memory reset (MRES) of the CPU via the mode switch


RUN
STOP
MRES
1. Set mode switch to STOP
2. Keep mode switch pressed in MRES position until RUN/STOP LED has flashed slowly twice
Then release again
within 3 s !!!
3. Keep mode switch pressed in MRES position until RUN/STOP LED starts to flash rapidly
4. Set mode switch to RUN
restart of CPU is performed
→ Reset to factory settings
RUN/STOP LED of S7-1500
SIEMENS
SIMATIC S7-1500
STOP
Result:
when PROGRAM card is inserted
→ Memory reset
when card is not inserted

Specific information about memory reset (MRES) of CPU via mode switch:

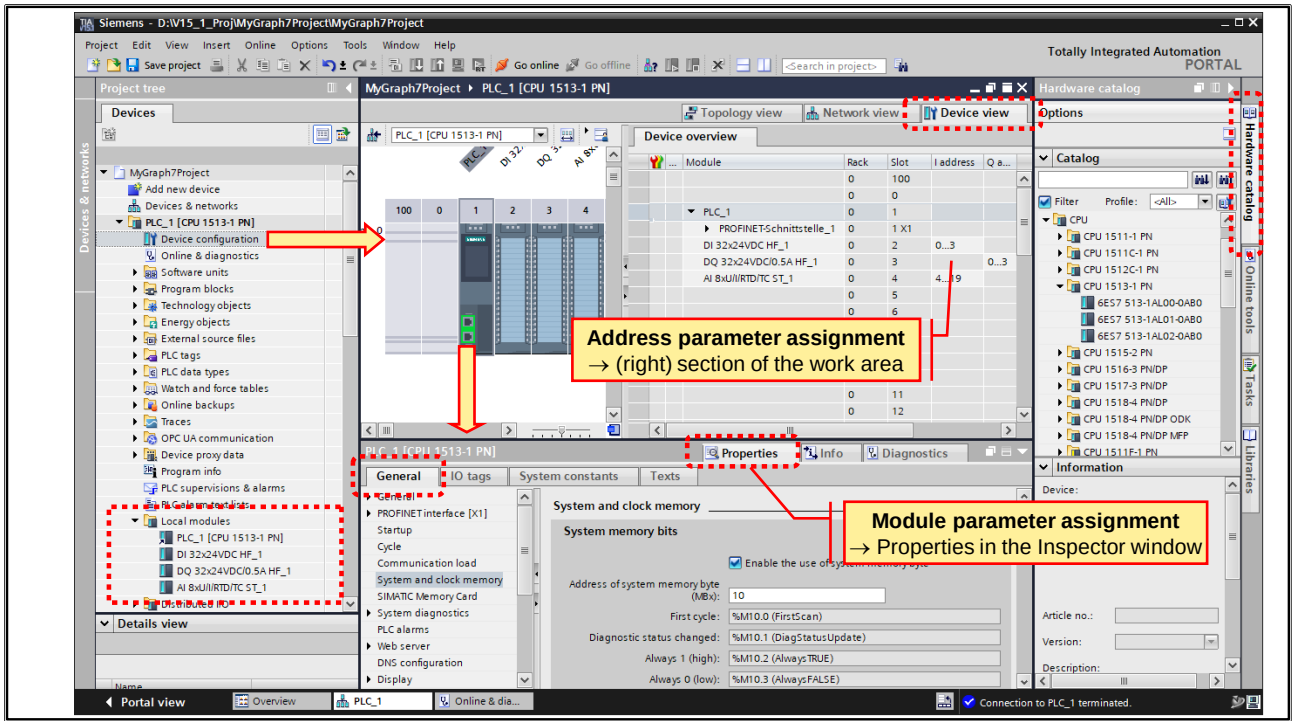
- **When memory card (MC) inserted => Memory reset**
  - Deletion of all user data (work memory, retentive memory)
    (process image, bit memory, counters, timers, program/data blocks)
  - Retained: Parameter assignment of the X1 interface, retentive part of the diagnostics buffer
  - The CPU copies all runtime-relevant data of the load memory (memory card) to the internal RAM work memory (runtime-relevant data: device configuration, program blocks, data blocks).

- **When no memory card (MC) inserted => Reset to factory settings**
  - Erasing of all memory areas of the CPU (work memory, retentive memory, diagnostics buffer, time) and IP address.

    After insertion of the MC, the runtime-relevant data of the load memory is downloaded from the memory card to the internal RAM work memory again.

    Device configuration (with IP address), program blocks, data blocks

2.11. Devices & networks: Device view (hardware configuration)


Address parameter assignment
→ (right) section of the work area
Module parameter assignment
→ Properties in the Inspector window

Creating/editing the hardware configuration

- "Devices & networks" editor → Select module → Select "Device view" tab
- Open the "Device configuration" editor of the module from the project tree

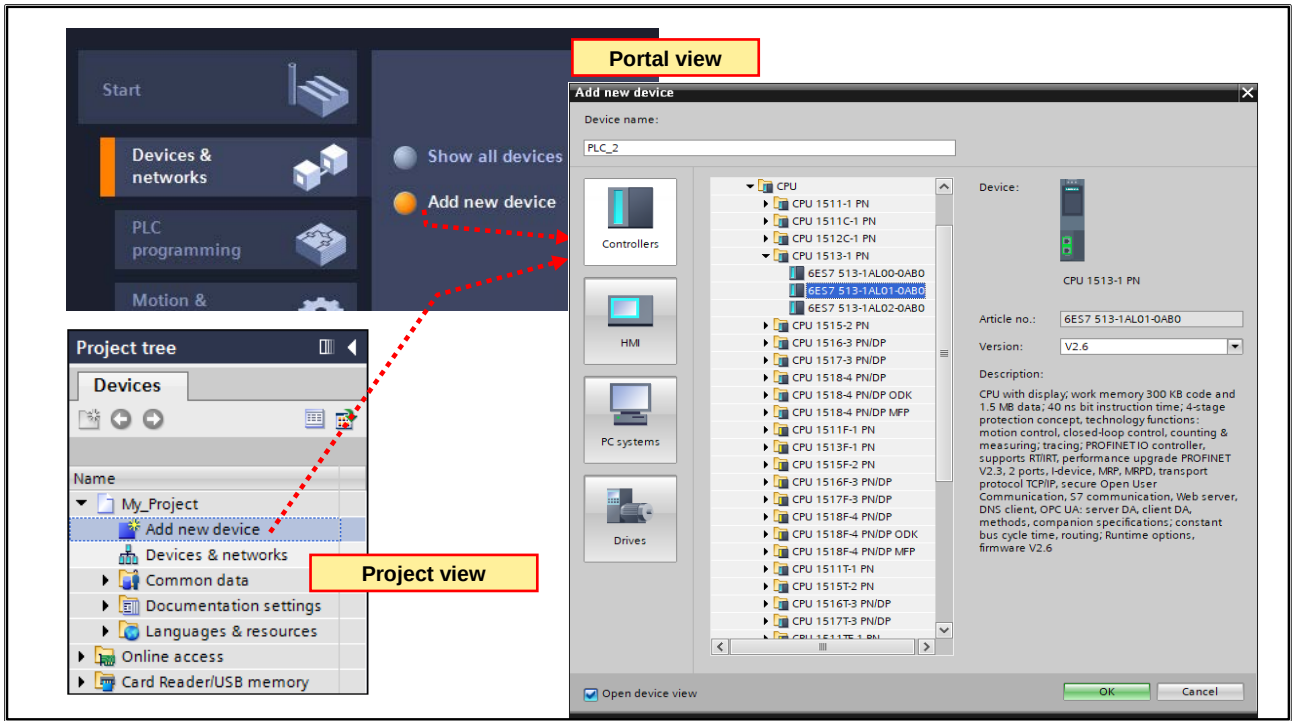
Components of the "Device configuration" editor

- **"Device overview"** section in the work area
  This editor consists of 2 sections. The lower section can be opened/closed using the buttons, and the split between the two sections can be repositioned as desired with the mouse.
  - Upper area = Module configuration
  - Lower area = Address parameter assignment of the configured modules
- **"Properties" tab** in the Inspector window
  This tab is used for parameter assignment of the module selected in the work area. All properties or parameters of the selected module are displayed and editable here. The left part of the Properties tab contains the area navigation in which the parameters are arranged in groups.
- **"Hardware catalog" task card**
  Module catalog for the configuration in the work area

Project tree → "Local modules"

The modules with their parameter assignment (e.g. addresses) are stored in the "Local modules" folder below the corresponding device in the project tree.

2.12. Preset configuration: Creating a hardware station



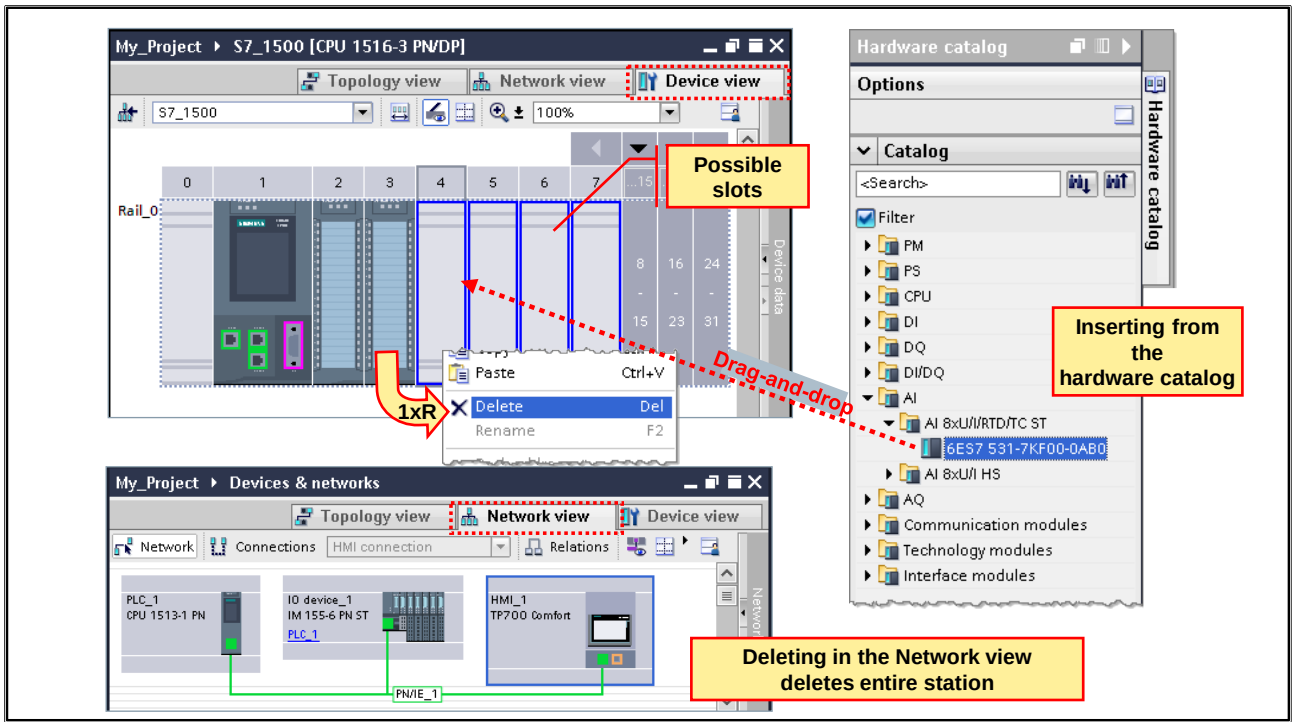
Device configuration

With the device configuration, the user specifies the arrangement of the modules in the rack.

When a new device is created a compatible rack is also created. The selected device is inserted at the first permitted slot of the rack.

Regardless of the method selected, the added device is visible in the device view and network view of the "Devices & networks" editor.

2.12.1. Inserting/deleting a module



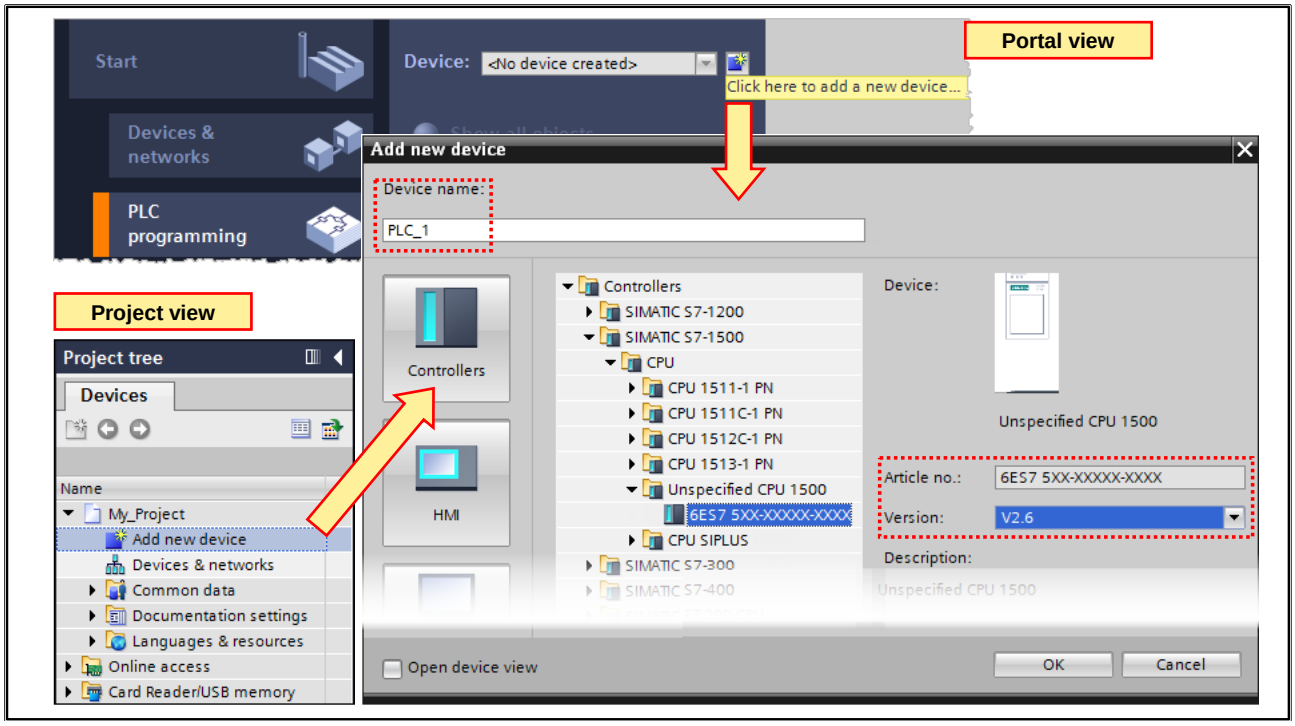
Inserting a module

A new module can also be inserted between previously inserted modules using drag-and-drop. To do so, the new module is dragged to the location above and between the two previously inserted modules. An insertion mark then appears, which can be used to insert the new module. All previously inserted modules are moved one slot to the right, and any redundant modules are moved to the "Modules not plugged in" area.

Deleting a module

Deleted hardware components are removed from the system and assigned addresses made available again.

2.13. Downloading actual configuration to project (1): Inserting an unspecified CPU

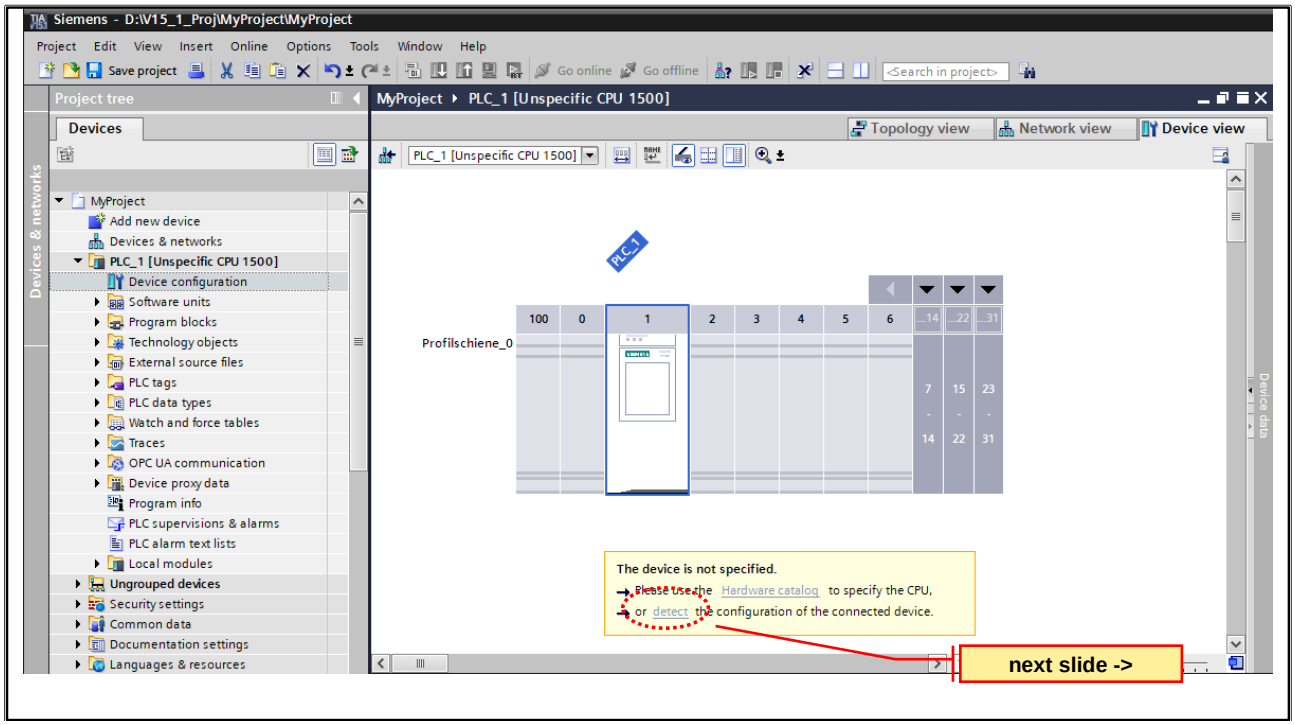


Uploading the actual configuration to the PG

The existing configuration (without module parameters!) can be read out from an actually existing station.

This is necessary, for example, when a matching offline project is not present on the PG. After the actual configuration is read out, the parameter assignment of the modules can be made, saved and downloaded back to the CPU. For this purpose, an "Unspecified CPU" must be created in the offline project in the first step.

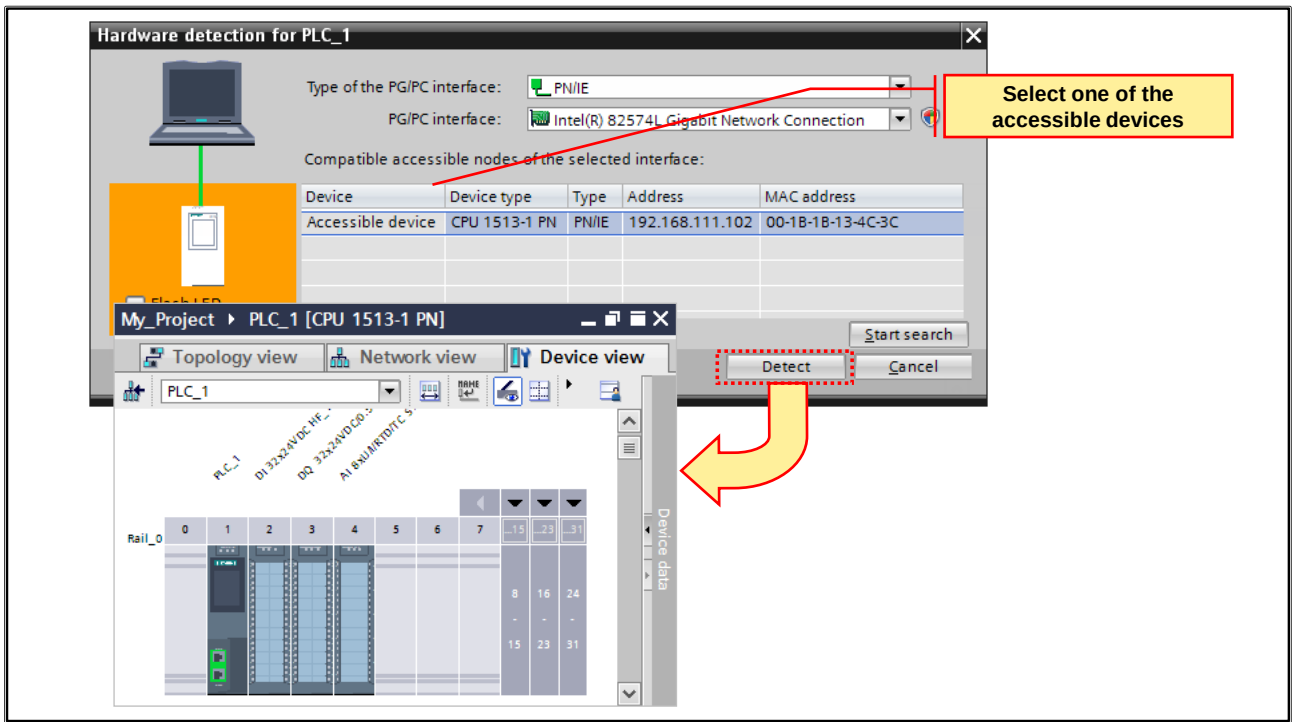
2.13.1. Downloading the actual configuration to the project (2): Determining accessible nodes



Determining the online station

As soon as the "Unspecified CPU" is created, all accessible devices can be determined in the displayed dialog using "detect".

2.13.2. Downloading the actual configuration to the project (3): Reading out the actual configuration



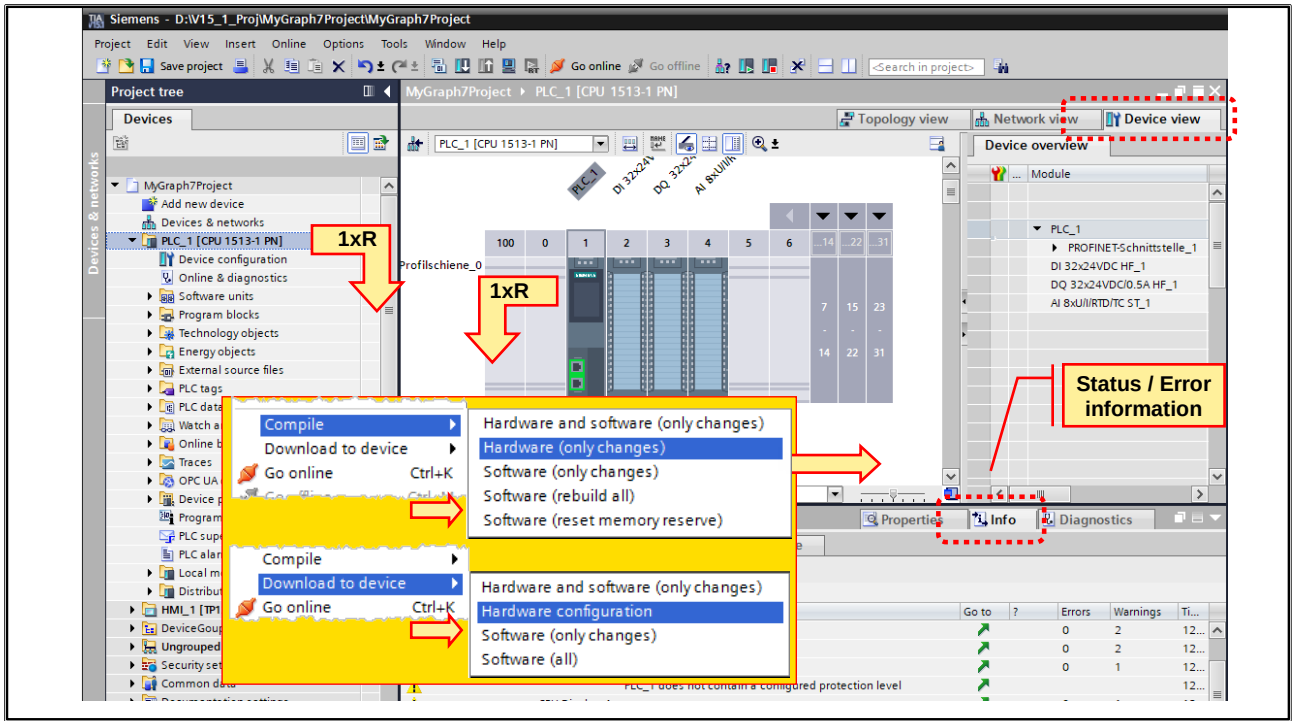
Downloading the actual configuration

The actual configuration of the selected node is read out and stored in the offline project.

Note:

When read out, only the hardware configuration is uploaded and not the hardware parameter assignment or S7 program blocks.

2.14. Compiling the hardware configuration and downloading it to the CPU

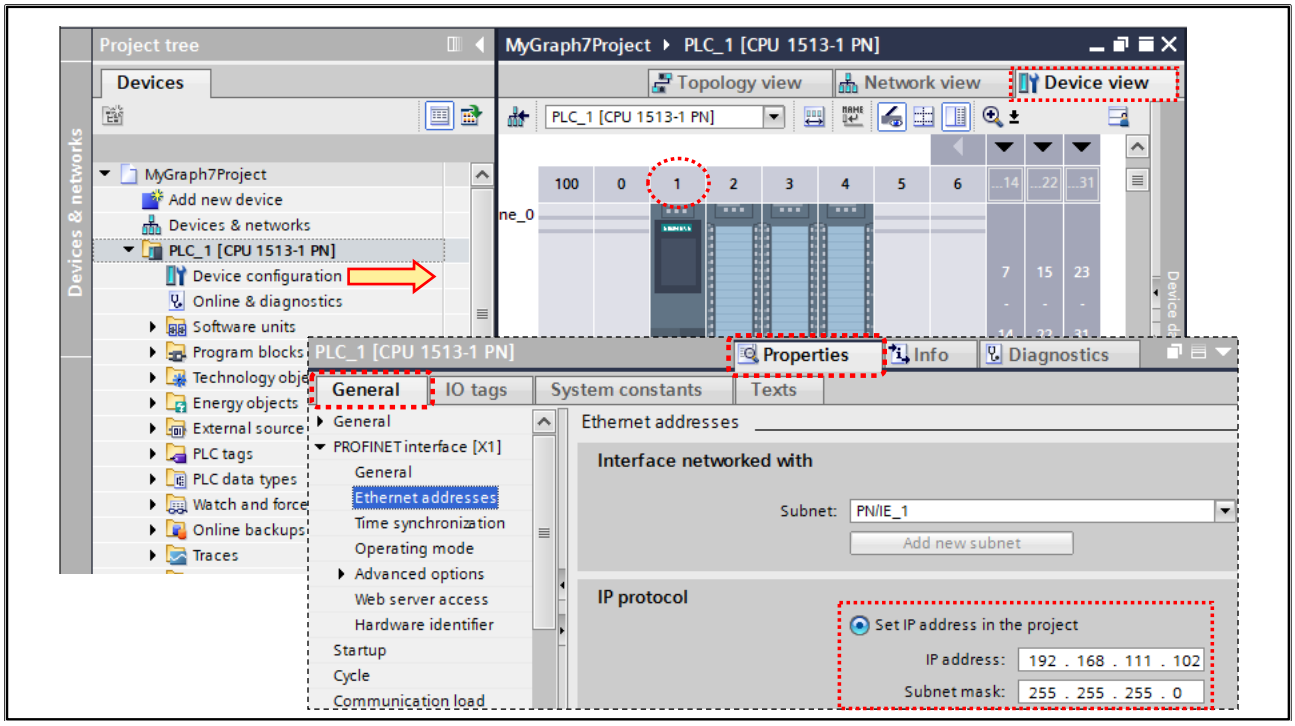


Compiling/downloading the hardware configuration

The following components of a hardware station can be compiled and downloaded:

- All
  The complete hardware configuration and parameter assignment and the complete user program are compiled/downloaded.
- Hardware/hardware configuration
  Only the complete hardware configuration and parameter assignment is compiled/downloaded.
- Software (only changes)
  Only the changed blocks of the user program are compiled/downloaded.
- Software (rebuild all)
  All blocks of the user program are compiled/downloaded.
- Software (reset memory reserve)
  All data blocks are reset to start values (download state). The data currently contained in the CPU is lost in this case, even if a memory reserve (for downloading a DB structure change with value preservation) is set in the properties of the data block.

2.15. CPU properties: Ethernet address

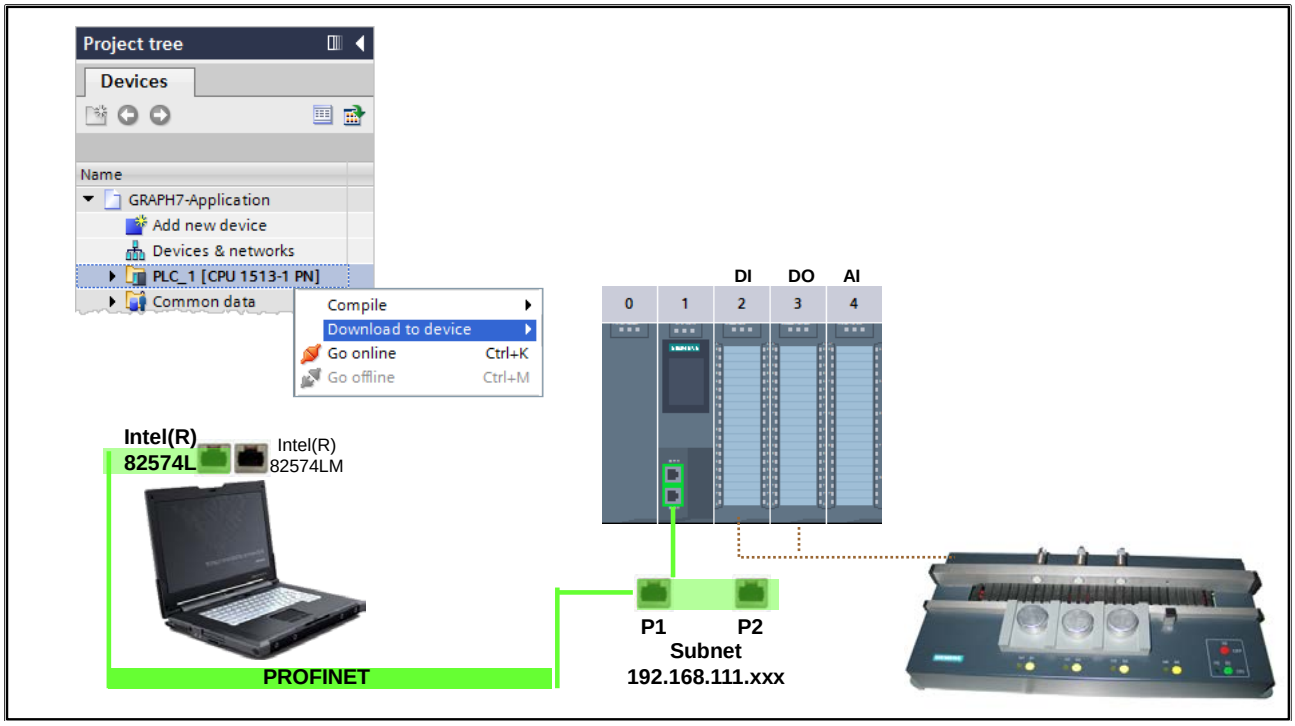


PROFINET interface

Regardless of whether the "Devices & networks" editor is open in the device view or network view, the settings of the CPU-PROFINET interface can be made for the selected CPU in the "Properties" tab of the Inspector window.

If an online connection between the programming device and CPU is to be established, both devices must have the same subnet mask and IP addresses that are in the same subnet.

2.16. Task: Hardware – Commissioning

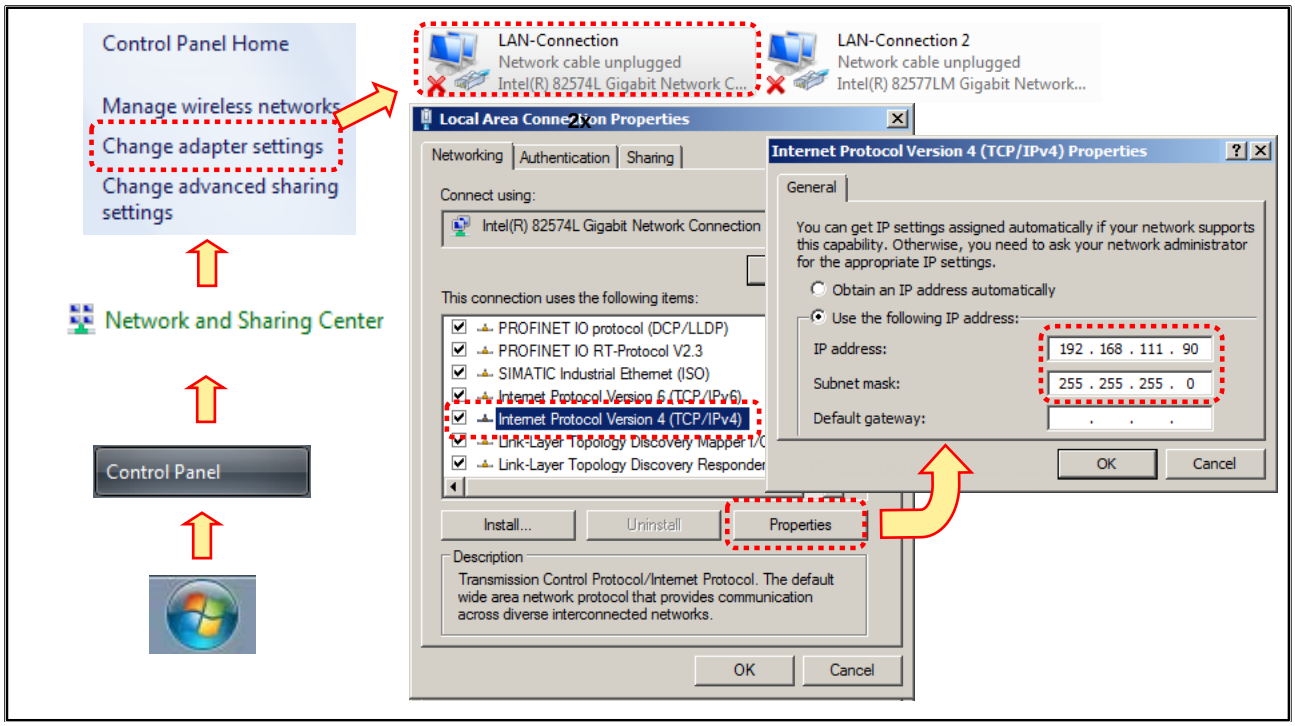


Note:

If you already have the necessary knowledge to create and parameterize a device configuration in the TIA Portal environment, this exercise can be skipped. The starting project "TIA_Graph_StartProject.zap15_1" can be used as the basis for all further exercises. The project already contains a finished device configuration. The configuration only needs to be transferred to the automation system. (Chapter 2.16.6)

(The distributed PNIO device ET 200SP contained in the demo case is optional.) The modules of the ET 200SP are not explicitly required for the duration of the course.)

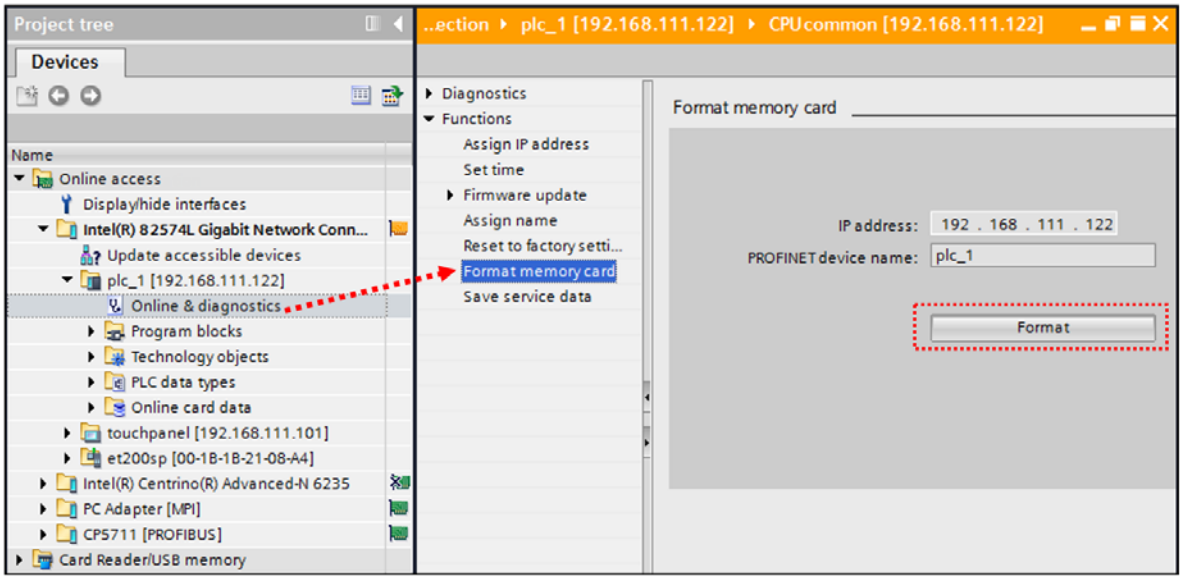
2.16.1. Exercise 1: Connect PG and set IP address of the PG



Task and procedure:

1. Connect the Ethernet interface "Intel(R) 82574L" of the PG to connection "P1" on the training device using an Ethernet cable.
2. Assign the IP address 192.168.111.90 and the subnet mask 255.255.255.0 to this PG interface. Follow the procedure shown in the figure for this.

2.16.2. Exercise 2: Erase SIMATIC memory card of the CPU



Task

To completely erase the CPU, the SIMATIC memory card of the CPU must also be erased. This can be done with:

- Windows-Explorer (SD card inserted in card reader of the PG)
- TIA Portal (SD card inserted in card reader of the PG)
- TIA Portal (SD card inserted in the CPU)

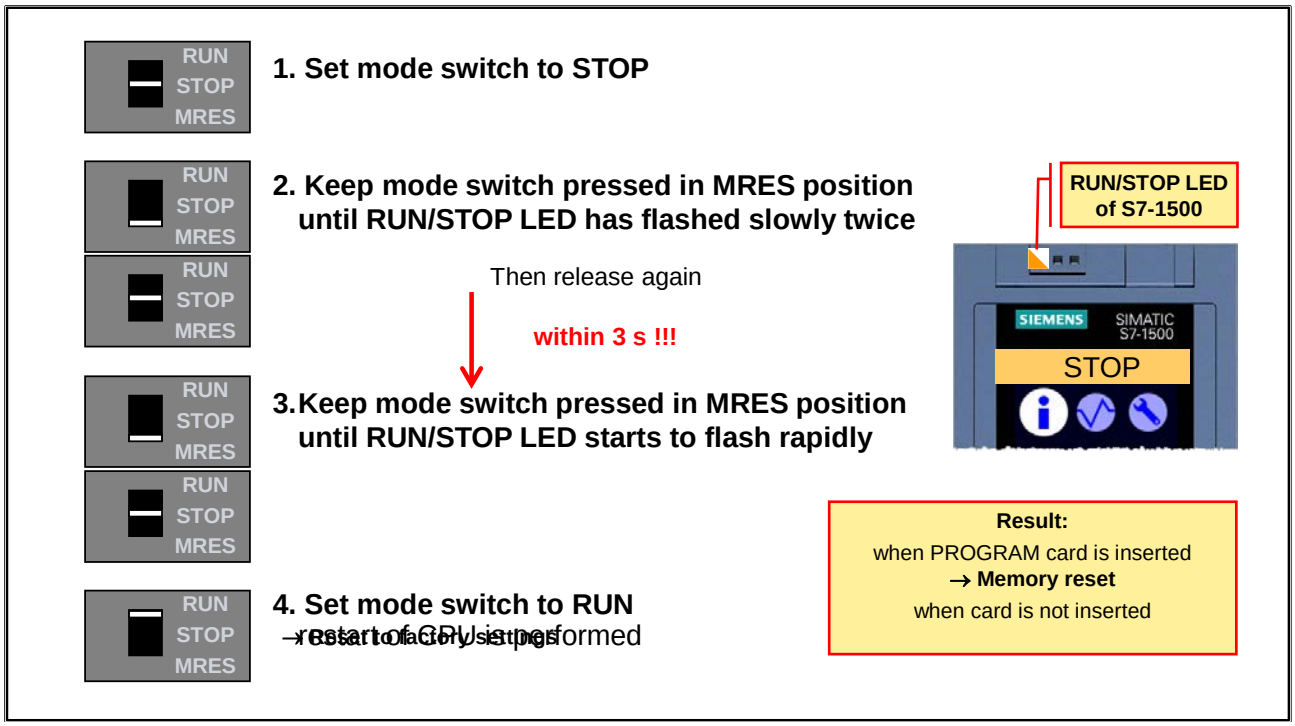
Procedure:

1. Check whether the SD card is inserted in the CPU.
2. Use "Display accessible nodes" in the Project view under the "Intel(R) 82574L" interface.
3. Select "Online & diagnostics" below the S7-1500 station (see figure).
4. Select "Format memory card" under "Functions" (see figure).

Note

If a password that is unknown to you is stored on the CPU, the SD card can only be erased if the SD is inserted in the card reader of the PG.

2.16.3. Exercise 3: Reset the CPU via the mode switch


RUN
STOP
MRES
1. Set mode switch to STOP
2. Keep mode switch pressed in MRES position until RUN/STOP LED has flashed slowly twice
Then release again
within 3 s !!!
3. Keep mode switch pressed in MRES position until RUN/STOP LED starts to flash rapidly
4. Set mode switch to RUN
→ restart of CPU is performed
RUN/STOP LED of S7-1500
SIEMENS
SIMATIC S7-1500
STOP
Result:
when PROGRAM card is inserted
→ Memory reset
when card is not inserted

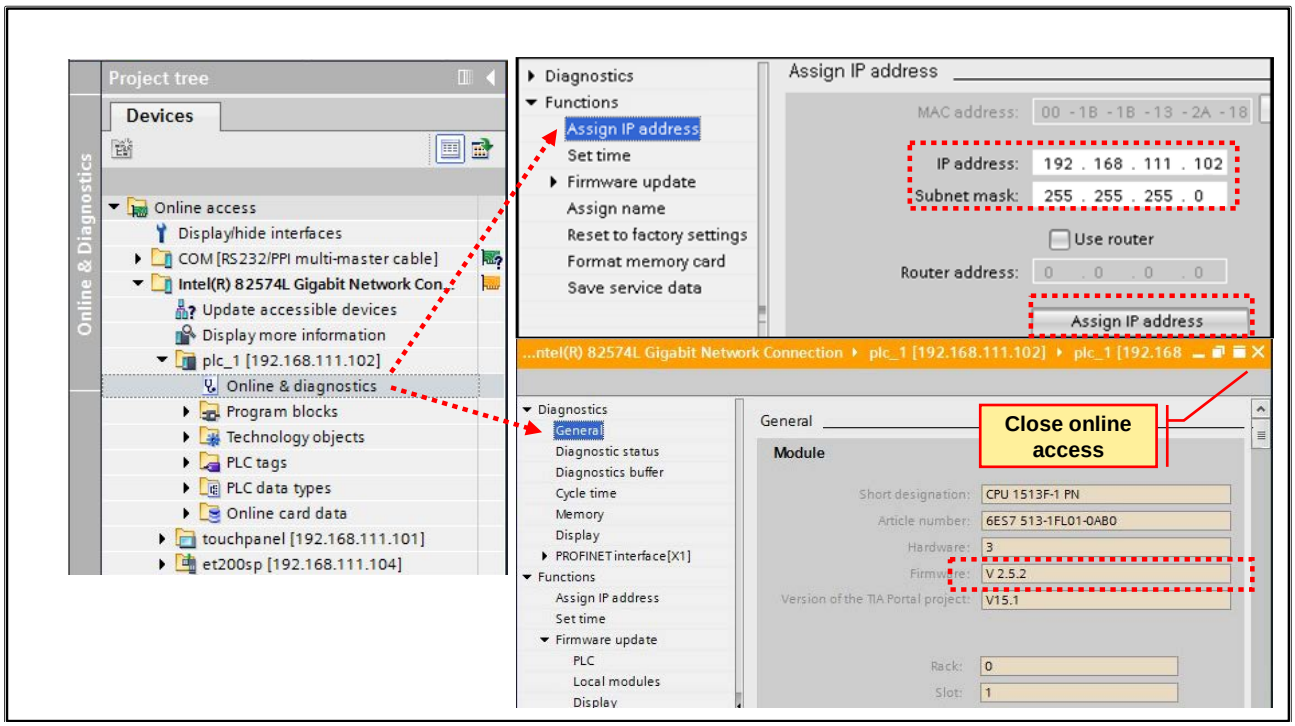
Task

Now that the SD card has been erased in the last exercise, the work memory of the CPU is now also to be erased. A memory reset must be performed for this.

Procedure

1. Perform the memory reset directly on the CPU according to the steps shown in the figure.
2. Perform a restart of the CPU by switching the mode switch from STOP to RUN.

2.16.4. Exercise 4: Determine CPU firmware version and assign IP address (node initiation)



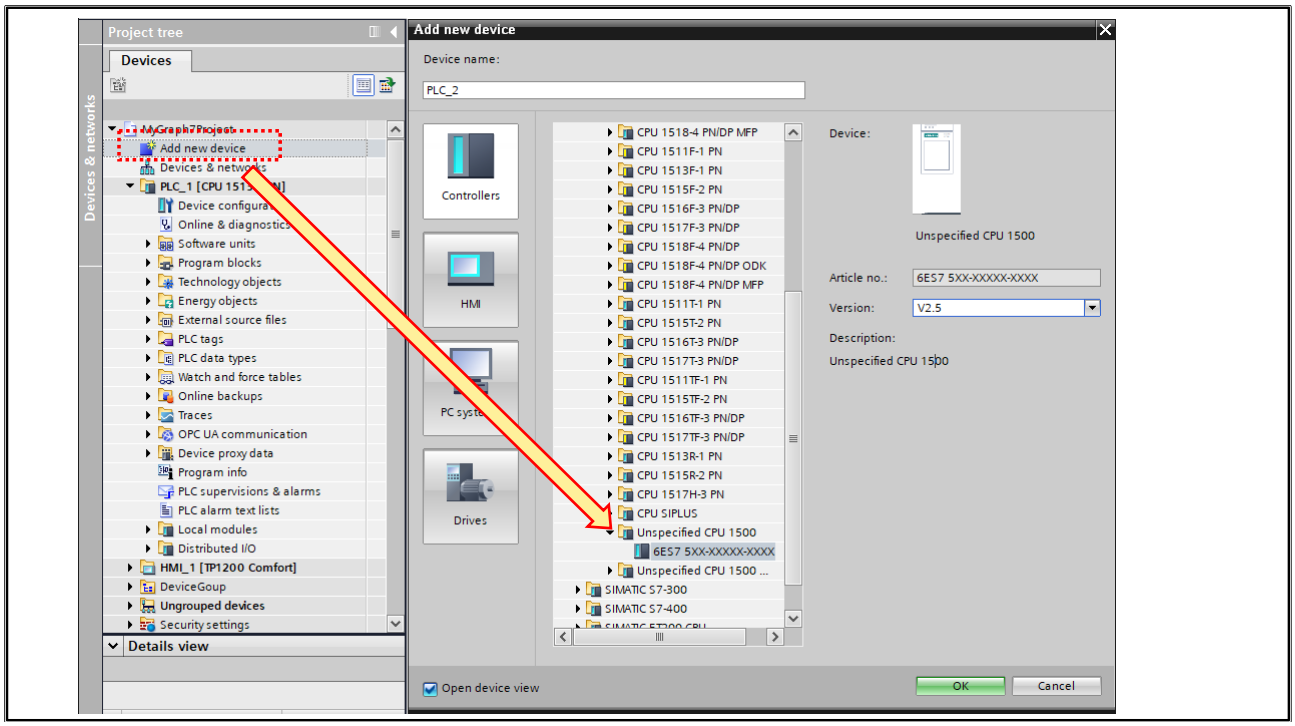
Task

In order to download, for example, S7 blocks from an offline project of an S7-1500 station to the online accessible CPU, the IP address of the CPU in the offline project must match that of the online accessible CPU. For this reason, you are to assign an IP address here to the online accessible CPU that you will later also assign in the CPU created in the offline project.

Procedure:

1. Use "Display accessible nodes" in the project tree under the "Intel(R) 82574L" interface
2. Select "Online & diagnostics" below the S7-1500 station (see figure)
3. Check and, if necessary, correct the IP address under "Functions --> Assign IP address"
4. Close online access in the work area (see figure)

2.16.5. Exercise 5: Configure S7-1500 station



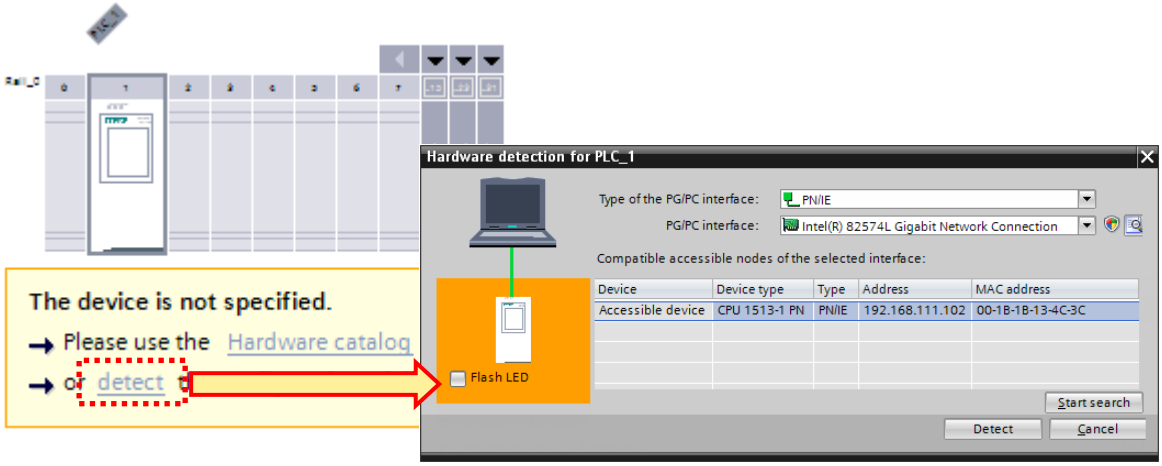
Task

You are to configure the S7-1500 station in such a way that the module configuration matches your actual training device.
Use the "Detect" hardware configuration function in the TIA Portal for this.

Procedure

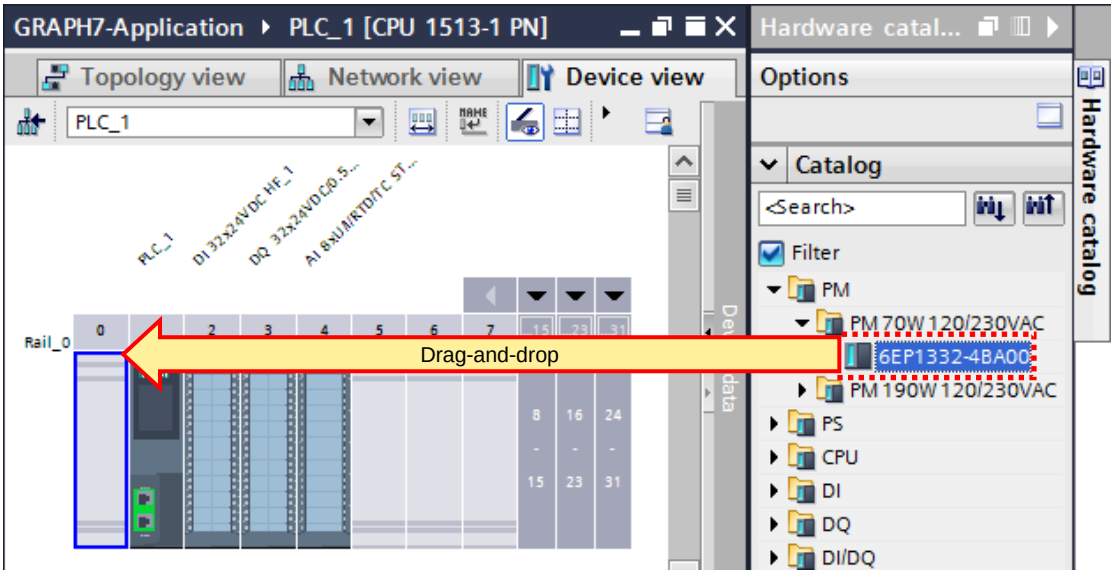
.

1. Create a project named GRAPH7 application in the TIA Portal.
2. Create an "Unspecified CPU 1500" by double-clicking the "Add new device" function.
3. "Detect" the actual configuration of the existing station.



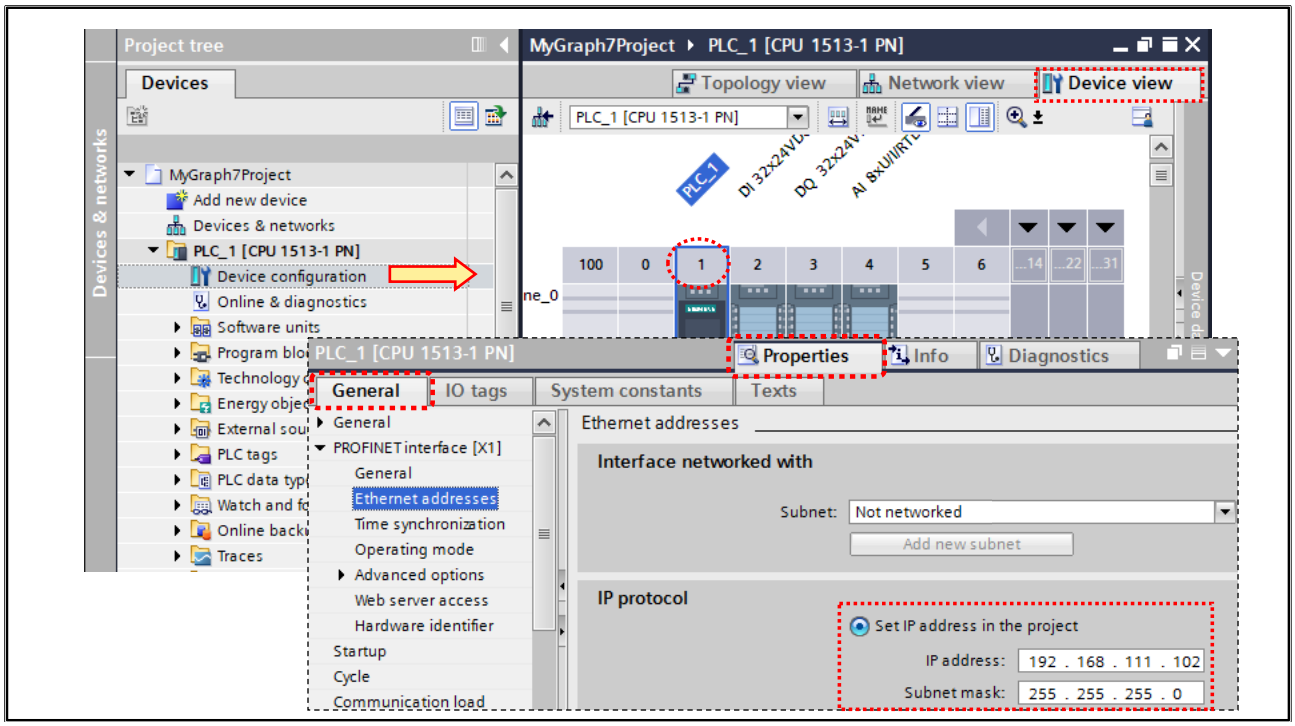
4. Use drag-and-drop to add the power module from the "Hardware catalog" task card to slot 0.

Get the module type and order number from the label of the modules of your training device or the created module overview.



5. Save your project.

2.16.6. Exercise 6: CPU properties: IP address and subnet mask



Task

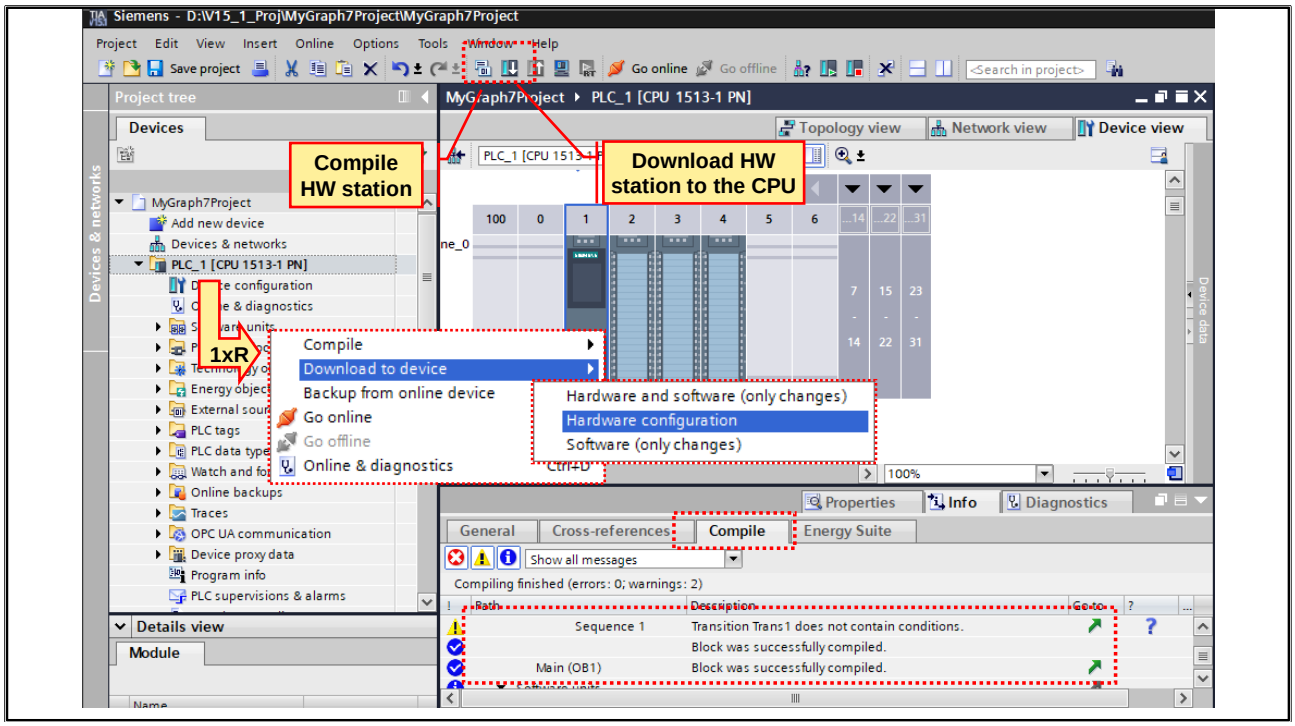
The S7-1500-CPU is to be assigned an IP address in the offline project.

A device name has no relevance for the PROFINET IO controller. This is only necessary for IO devices for device identification by the IO controller.

Procedure

1. Select the CPU in the "Device view".
2. Select the "Properties" tab of the Inspector window and select "Ethernet addresses". Enter the IP address and subnet mask shown in the figure.

2.16.7. Exercise 7: Compile the hardware configuration and download it to the CPU



Task

The configuration and parameter assignment of the S7-1500 hardware station is to be compiled and downloaded to the CPU.

Note:

As long as the CPU contains no program, the CPU does not switch to RUN state following a restart. In other words, if you download only the hardware configuration to the CPU in this exercise, as shown in the figure, the CPU will not change to RUN state at a subsequent restart.

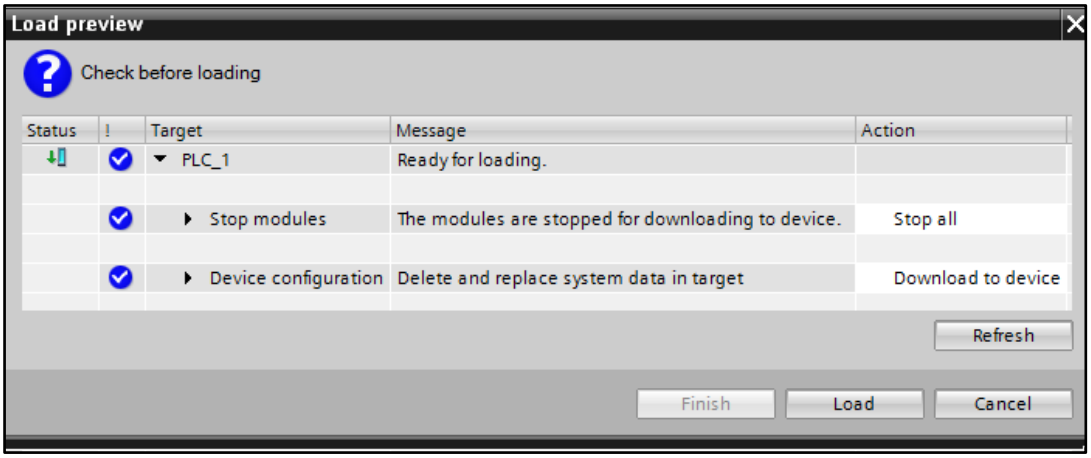
Procedure

1. Select your S7-1500 station in the Project view
2. Compile the HW station (right-click the station, see figure)
3. After error-free compilation, download the hardware configuration to the CPU (right-click the station, see figure)

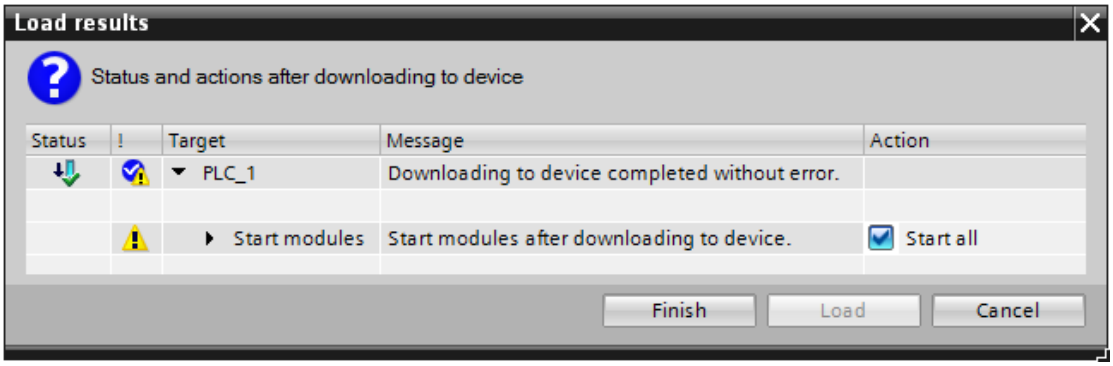
Note: When the adjacent buttons are used, the entire station, i.e. hardware and software, is always compiled or downloaded.

Continued on next page

4. Fill out the dialog that appears as follows and close it with "Load":

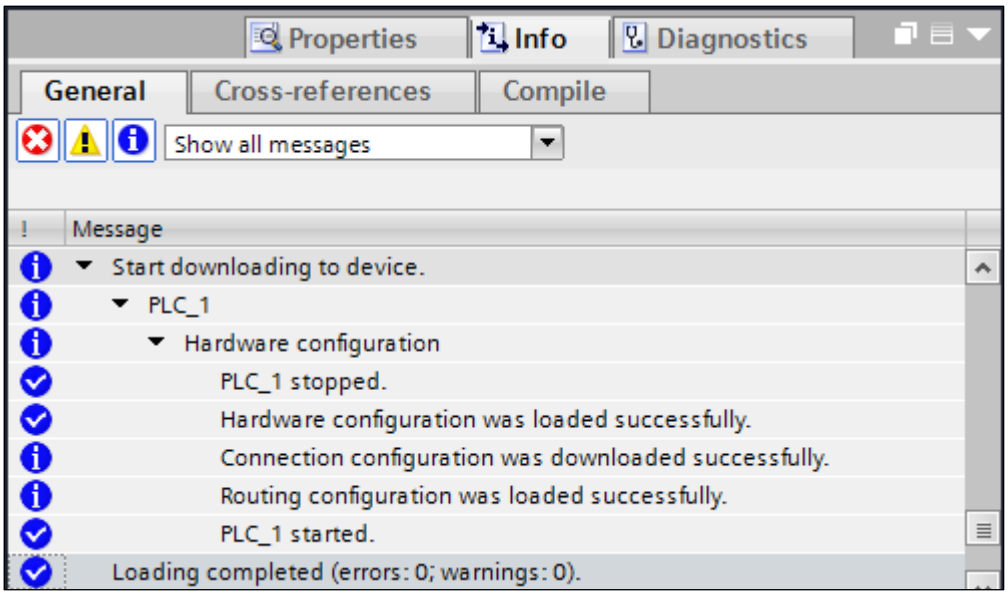


5. Once you have clicked "Load", another dialog appears that you fill out as follows and close with "Finish".



If you have downloaded only the hardware configuration to the CPU, the CPU would now have to remain in STOP state.

6. Check whether the hardware configuration was successfully downloaded in the "Inspector window" under "Info -> General".



7. Save your project and finish the exercise.

Table of contents

3

3. Getting Started with S7-GRAPH 3-2

3.1. Programming sequential functional sequences 3-3
3.2. Sequencer: Representation according to IEC 61131-3 3-4
3.3. Basic elements of a sequencer: Step and transition 3-5
3.4. Other elements of the sequencer 3-6
3.5. GRAPH: Creating a block 3-7
3.6. Sequence view of a step sequence 3-8
3.7. Simple standard actions in the step 3-9
3.8. Transitions for sequential control 3-10
3.9. Testing a step sequence 3-11
3.10. Permanent instructions 3-12
3.11. Processing order of GRAPH FB 3-13
3.12. Settings for the GRAPH editor and compiler 3-14
3.13. Call of a GRAPH FB 3-15
3.14. Adapting maximum interface parameters subsequently 3-16
3.15. General information about libraries 3-17
3.16. Task description 3-18
3.17. Exercise 1: Parameter assignment of the S7-1500 CPU 3-20
3.18. Exercise 2: Configure a linear sequencer 3-21
3.19. Exercise 3: Call up of the GRAPH FB in OB1 3-22
3.20. If you want to know more 3-23
3.1.1. Simulation of PLC and HMI devices 3-24
3.1.2. Setting the S7 online interface for the simulation 3-25

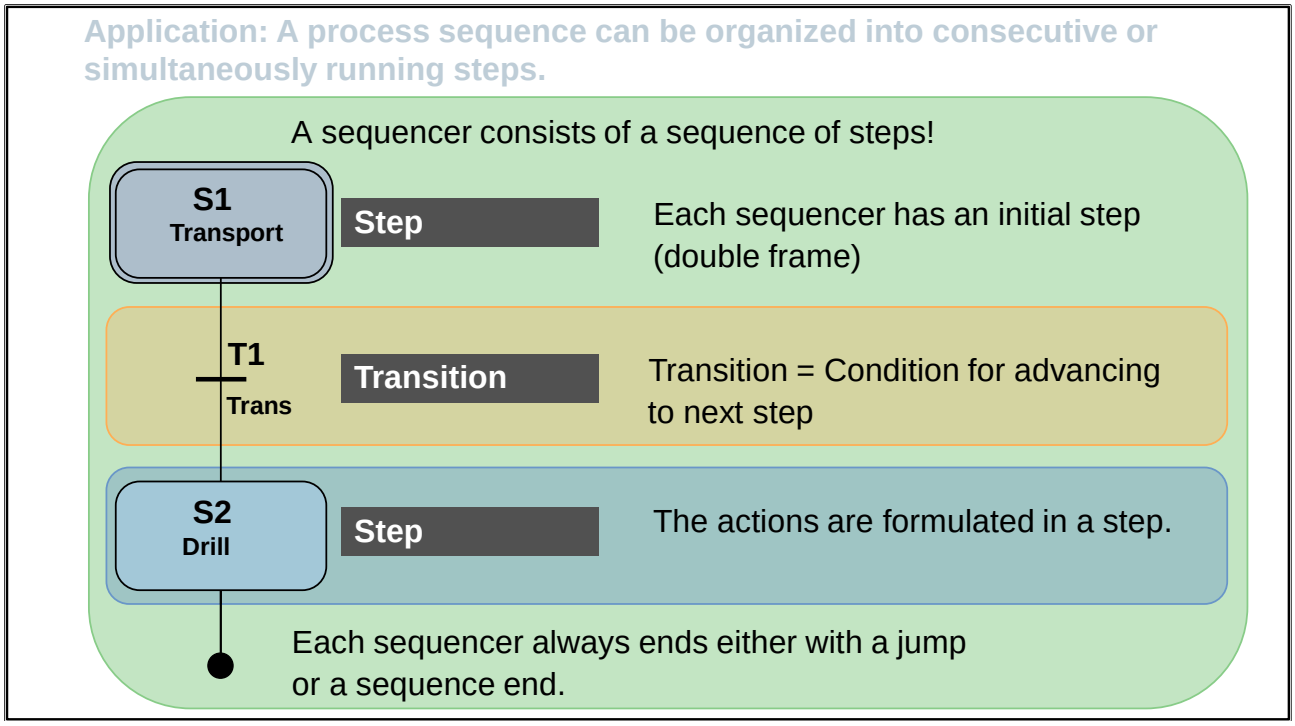
3. Getting Started with S7-GRAPH

The participant should ...

… know the essential elements of a step sequence

... be able to program a linear sequencer

3.1. Programming sequential functional sequences


Application: A process sequence can be organized into consecutive or simultaneously running steps.
A sequencer consists of a sequence of steps!
S1
Transport
Step
Each sequencer has an initial step (double frame)
T1
Trans
Transition
Transition = Condition for advancing to next step
S2
Drill
Step
The actions are formulated in a step.
Each sequencer always ends either with a jump or a sequence end.

GRAPH

The GRAPH programming language is used to control sequential processes. The process is hereby broken down into individual steps with their own functional scope. The sequence is graphically represented and can be documented with images and text.
The actions to be executed are defined within the individual steps, and the step-enabling conditions govern the transitions. Their definitions, interlocks, and supervisions are specified using a subset of STEP 7 programming languages LAD and FBD.

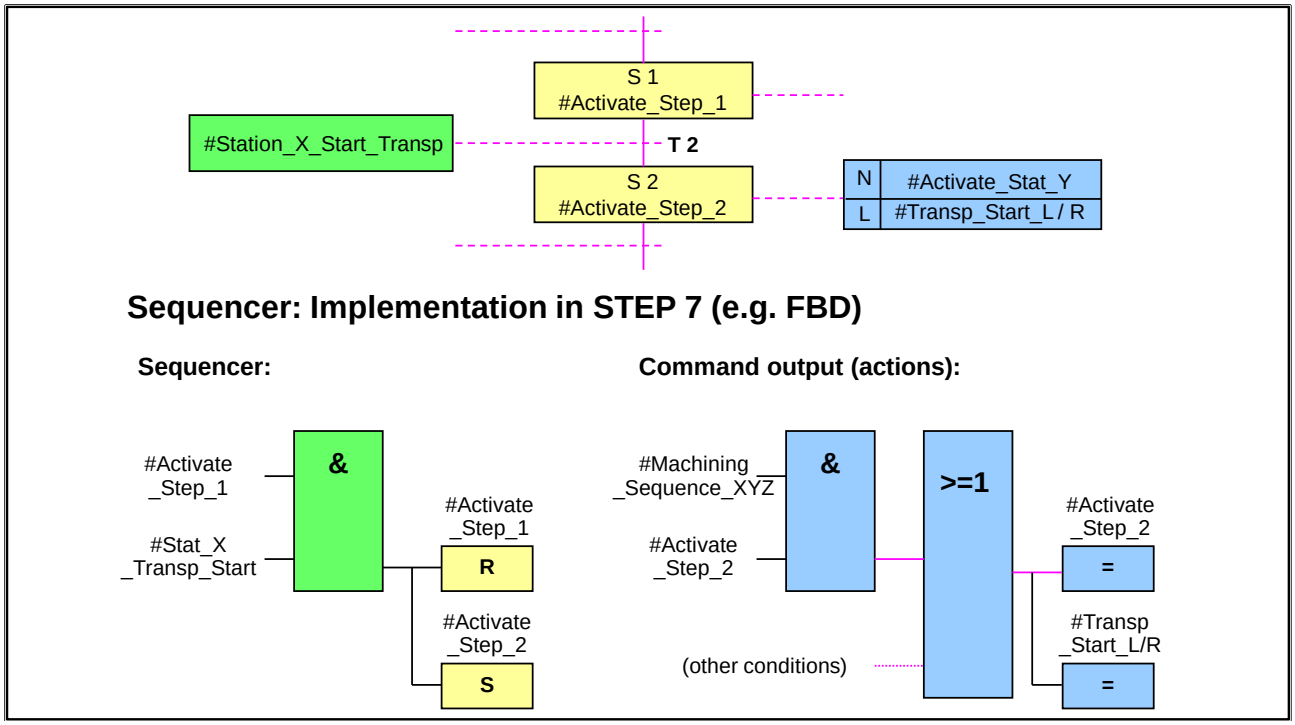
GRAPH for S7-1500 is compatible with the sequential function chart language specified in IEC 61131-3.

Sequencer

Steps and transitions form a sequencer. The sequencer is stored in an FB. An instance DB that contains the data of the sequencer is assigned to the FB. It is therefore not possible to save the sequencer in an FC!
An executable program requires at least three blocks:

- The FB that contains the sequencer(s)
- An instance DB with the data of the sequencer
- An organization block, a function block or a function with the FB call

3.2. Sequencer: Representation according to IEC 61131-3



Transitions

Transitions are step-enabling conditions through which the transition or advancement from one step to the next occurs. Step-enabling conditions can be process signals (e.g. signals of sensors, position switches, etc.) and/or result from logical conditions.

Step sequence

In the step sequence program section, the step tags are managed or the user specifies the desired order of the production process. A separate network is generally created per step tag.

A step tag is set if the predecessor step is active AND the corresponding transition to this step is true. The preceding step is simultaneously reset under the same conditions. By querying both the preceding step conditions and transition conditions, the unwanted activation of a step at the wrong time is prevented.

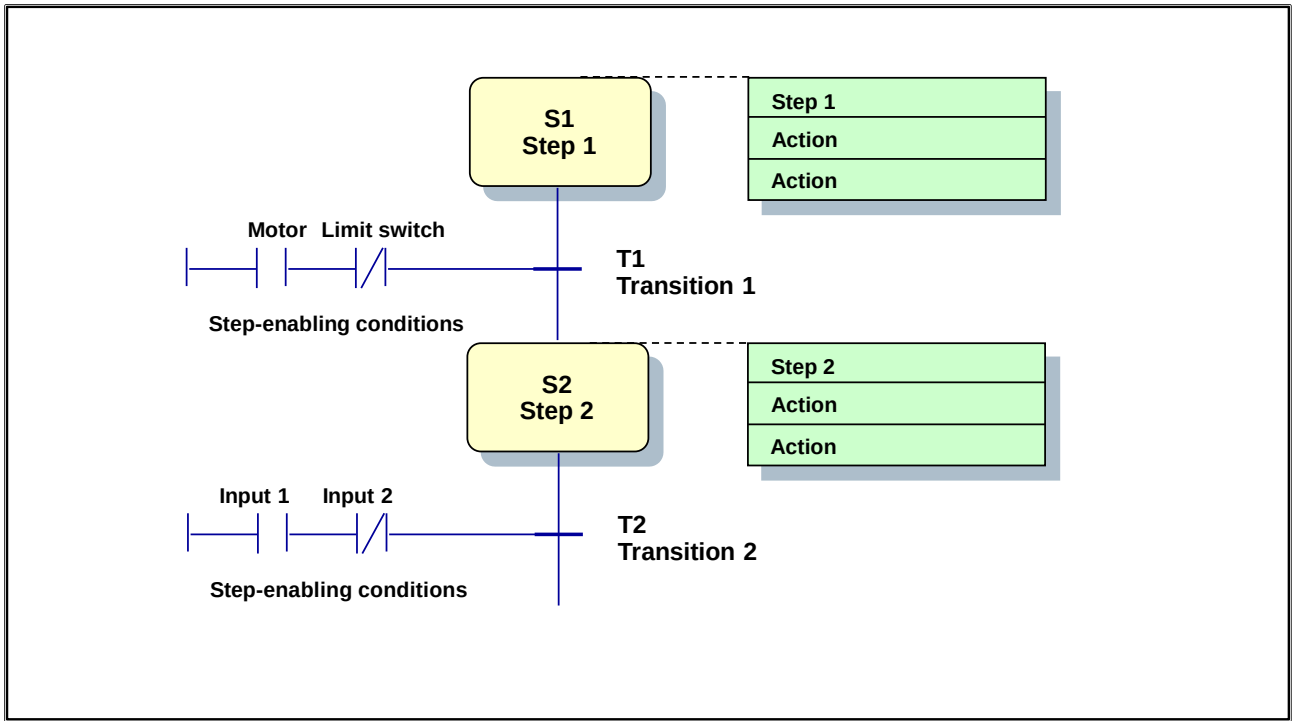
The step sequence program section is responsible exclusively for setting and resetting the step tags and not for controlling the final controlling elements of the process or for executing actions.

Command output

In the command output program section, the user specifies what is to occur in which step. One or more actions are assigned to each step, to which the following qualifiers can be added:

- D = delayed (time-delayed after activation of the step)
- N = not stored (non-latching, action is executed for only as long as the step is active)
- L = limited (time-limited, even if the step is active for a longer time)

3.3. Basic elements of a sequencer: Step and transition

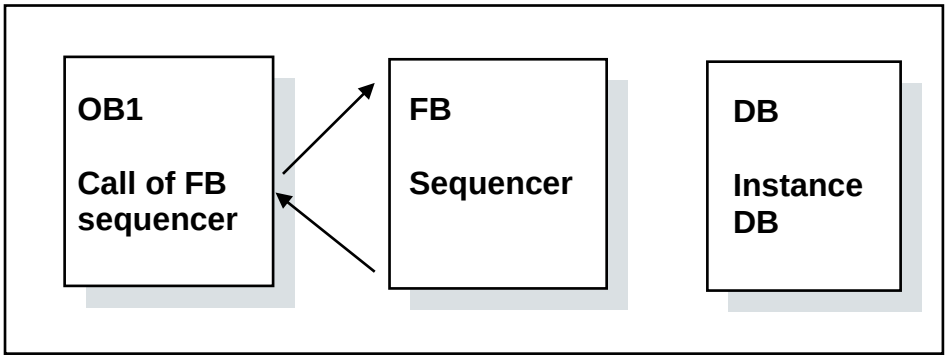


Sequencer

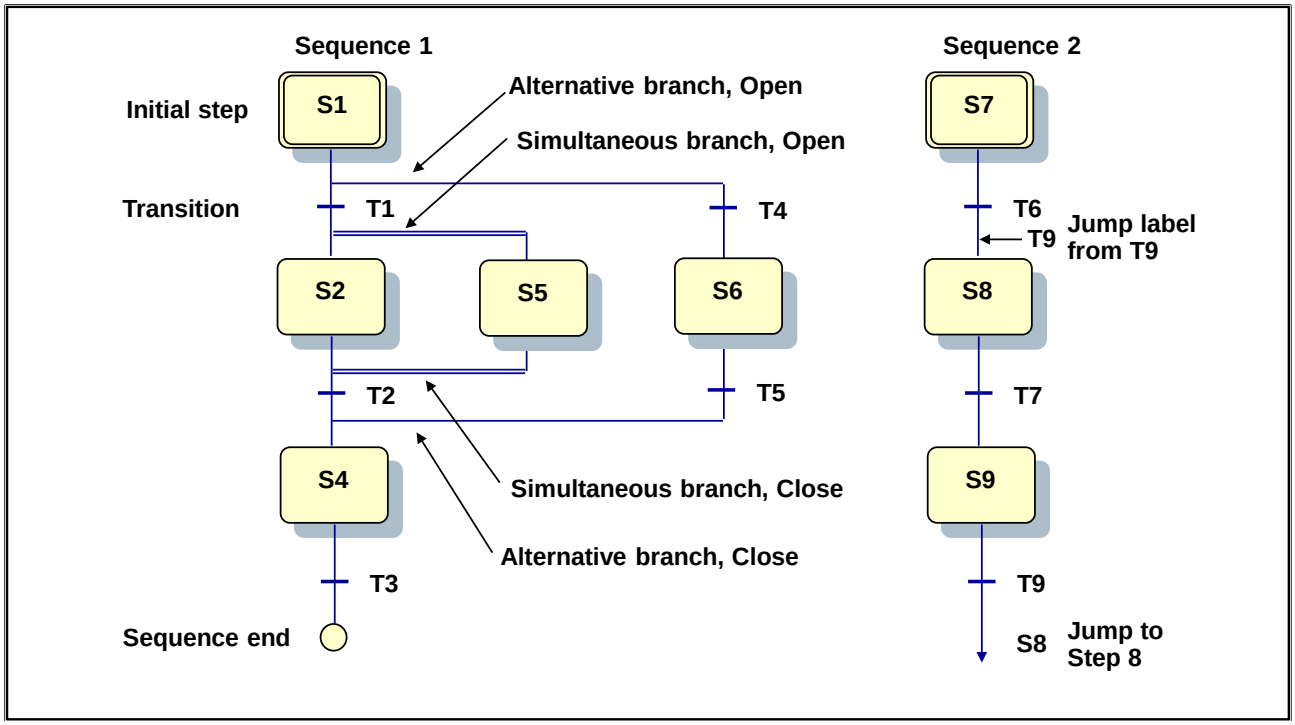
Steps and transitions form a sequencer. The sequencer must be stored in an FB. An instance DB that contains the data of the sequencer is assigned to the FB.

An executable program requires at least three blocks:

- The FB that contains the sequencer(s)
- An instance DB with the data of the sequencer
- An organization block, a function block or a function with the FB call



3.4. Other elements of the sequencer



Overview

When a sequencer is created, integrated software functions facilitate the graphic programming. A sequence structure can be easily created without programming expertise by arranging the GRAPH structure elements in such a way that a graphical representation of the sequencer results.

Structure

A sequencer consists of a sequence of steps and transitions. This sequence can be linear or branched.

- The commands to the plant (actions) are formulated in the steps.
- The transitions contain the conditions for the advancement from one step to the next.

The figure above shows the elementary elements that a sequencer can be composed of. These symbols can be selected from the toolbar.

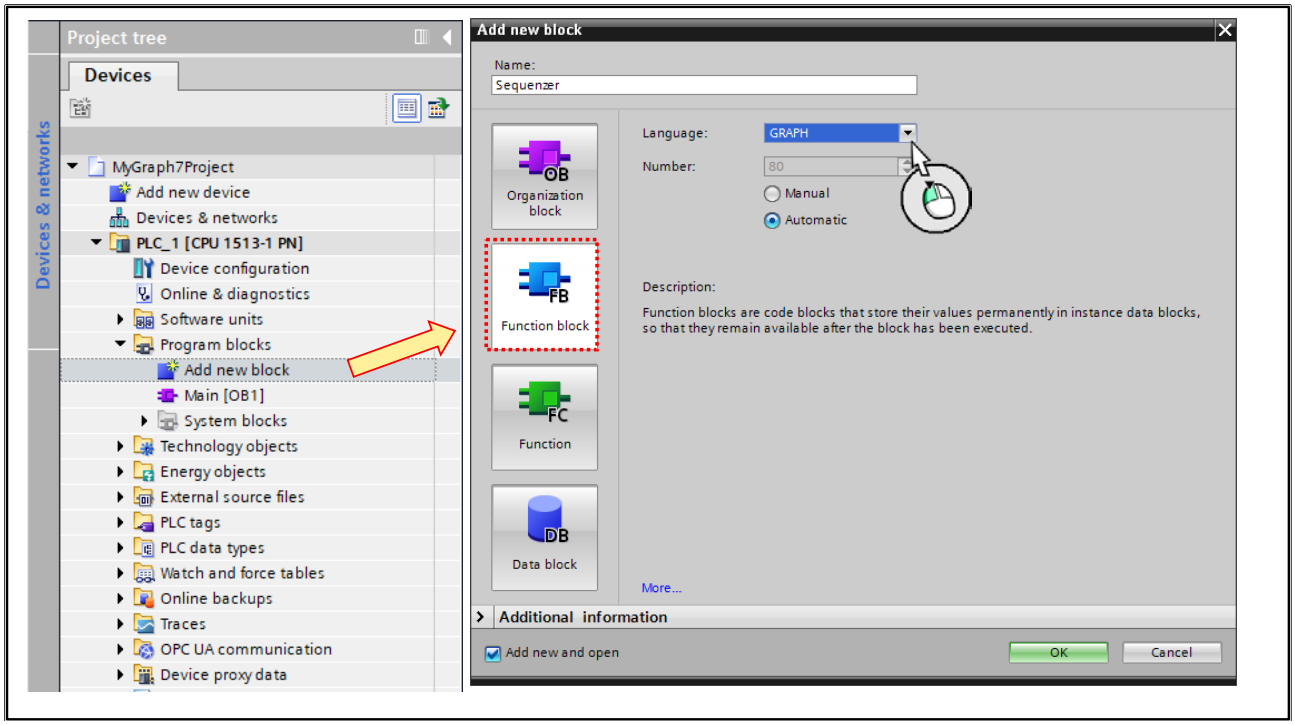
- Step and transition
- Alternative branch, OPEN and CLOSE in each case
- Simultaneous branch, OPEN and CLOSE in each case
- Jump
- Sequence end
- Insert sequence

In addition, there are symbols for permanent instructions and for switchover of insertion mode.

Editing

Selection of the element that is to be inserted using drag-and-drop. Selection of position with the mouse pointer at which the element is to be inserted. Small green rectangles show possible insertion positions.

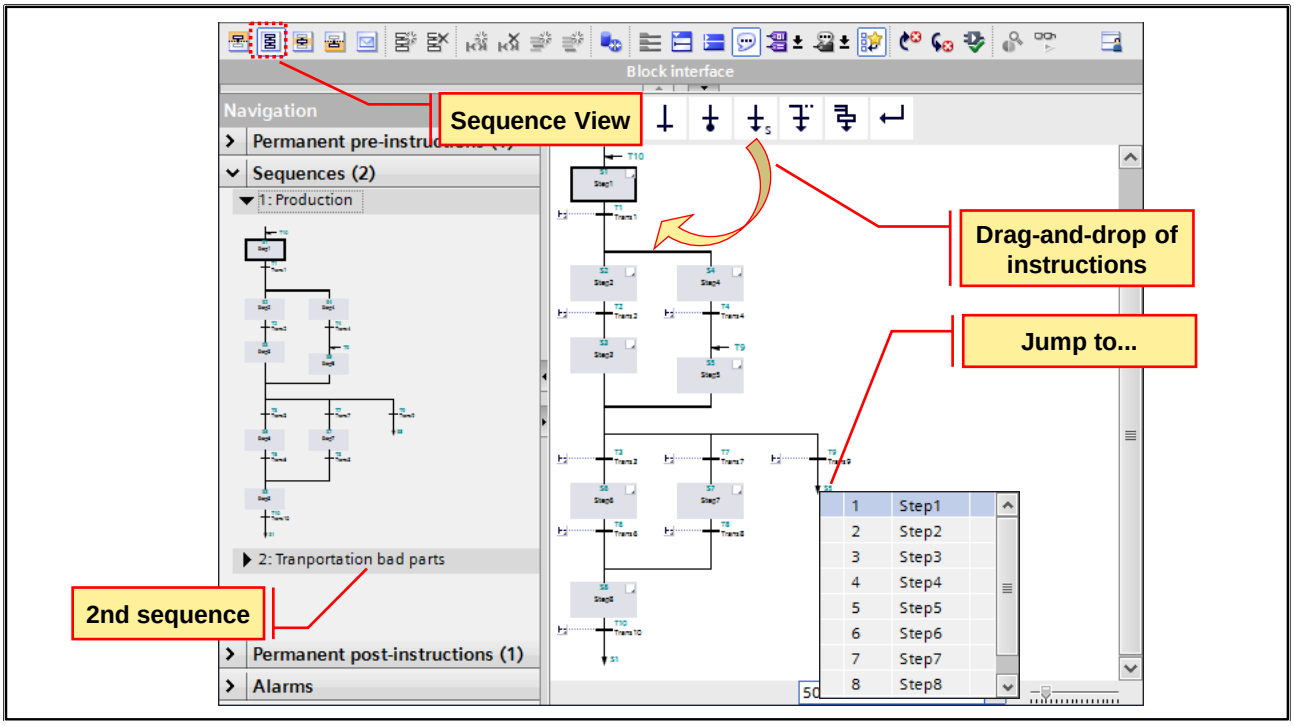
3.5. GRAPH: Creating a block



Creating a GRAPH block

GRAPH blocks are created just the same as LAD/FBD/SCL/STL blocks. It must be noted, however, that GRAPH blocks must be of block type FB and must not be incorporated in multi-instances.

3.6. Sequence view of a step sequence



Overview

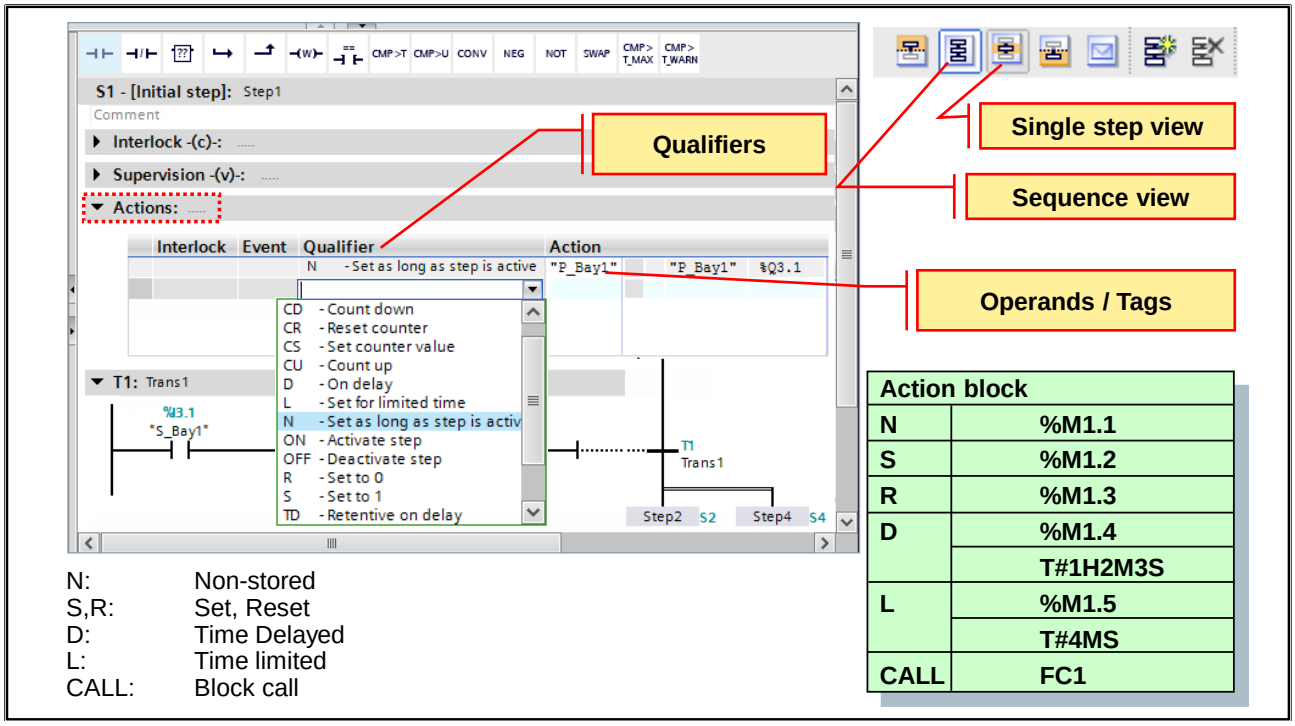
The sequence view of a step sequence displays an overview in the work area of all steps and transitions of a sequence that is being created. The content of the transitions and the actions of a step can be expanded and minimized.
An overview of all available sequences is displayed on the left in the navigation area.

Structure of a sequencer

A sequencer consists of a sequence of steps and transitions. This sequence can be linear or branched. The commands to the plant are formulated in the steps. The transitions contain the conditions for the advancement from one step to the next. The following symbols represent the elements that a sequencer can be composed of. You can select these symbols from the toolbar.

Symbol	Name	Shortcut
	Step and transition	SHIFT + F5
	Step	---
---	Transition	
	Sequence end	SHIFT + F7
	Jump	SHIFT + F12
	Open alternative branch	SHIFT + F8 (focus on step)
	Open simultaneous branch	SHIFT + F8 (focus on transition)
	Close branch	SHIFT + F9

3.7. Simple standard actions in the step



Action block	
N	%M1.1
S	%M1.2
R	%M1.3
D	%M1.4
	T#1H2M3S
L	%M1.5
	T#4MS
CALL	FC1

N: Non-stored
S,R: Set, Reset
D: Time Delayed
L: Time limited
CALL: Block call

Overview

These actions are all defined in IEC 61131-3 under 6.7.4.5 Action qualifiers:

D, L, N The associated actions are reset as soon as the step has finished.

D, L The times can be specified as a constant or a tag.

S, R The corresponding actions persist after the step.

CALL Block call FBi.DBi, FCi, SFBi.DBi, SFCi (i = block number):
Calls the specified block. After execution of the block, the step sequence program continues running.

Note

Operands are addressed using symbolic or absolute addresses in the action block. After input, absolute addresses are automatically assigned a new symbol or, if a symbol already exists for this address, this address is displayed.

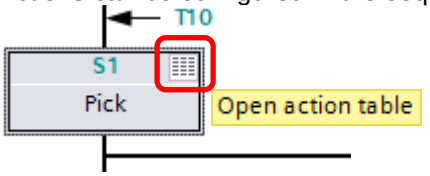
The actions in the step are executed according to the top-down principle! That is, an appropriate order of execution should be defined when the block is created.

Time information

Time information must be entered in IEC time format T# …, e.g. T#10s.

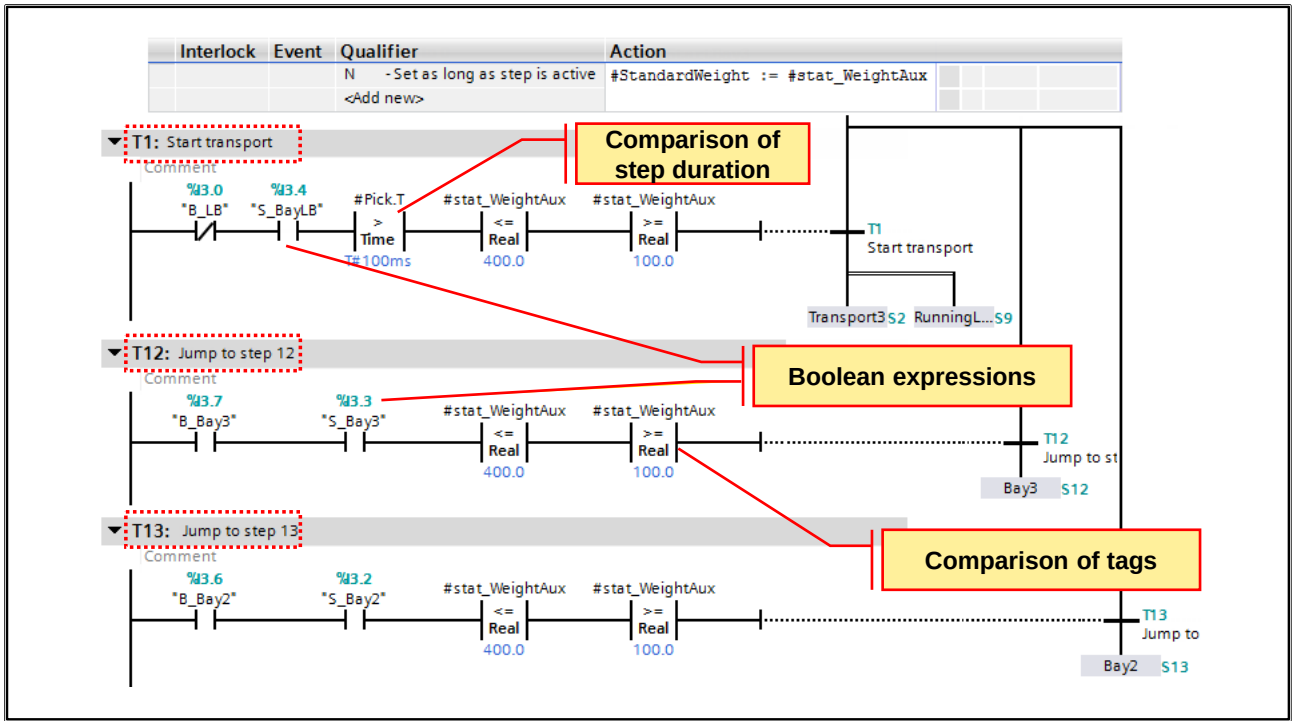
Input of actions

Actions can be configured in the sequence view after expanding the action table



or in the single step view.

3.8. Transitions for sequential control



Overview

These conditions, which are allowed in a transition, can all be found in IEC 61131-3 under 6.7.3 Transitions: Examples include:

- Boolean expressions such as & combinations
- Comparisons of step duration
- Comparisons of tag values
- Empty transitions that always enable the next step immediately

Note

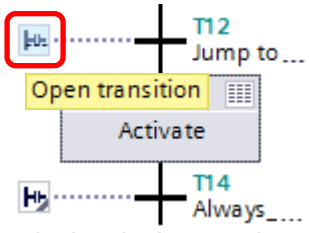
Operands are addressed using symbolic or absolute addresses in the action block. After input, absolute addresses are automatically assigned a new symbol or, if a symbol already exists for this address, this address is displayed.

Time information

Time information must be entered in IEC time format T# …, e.g. T#100ms.

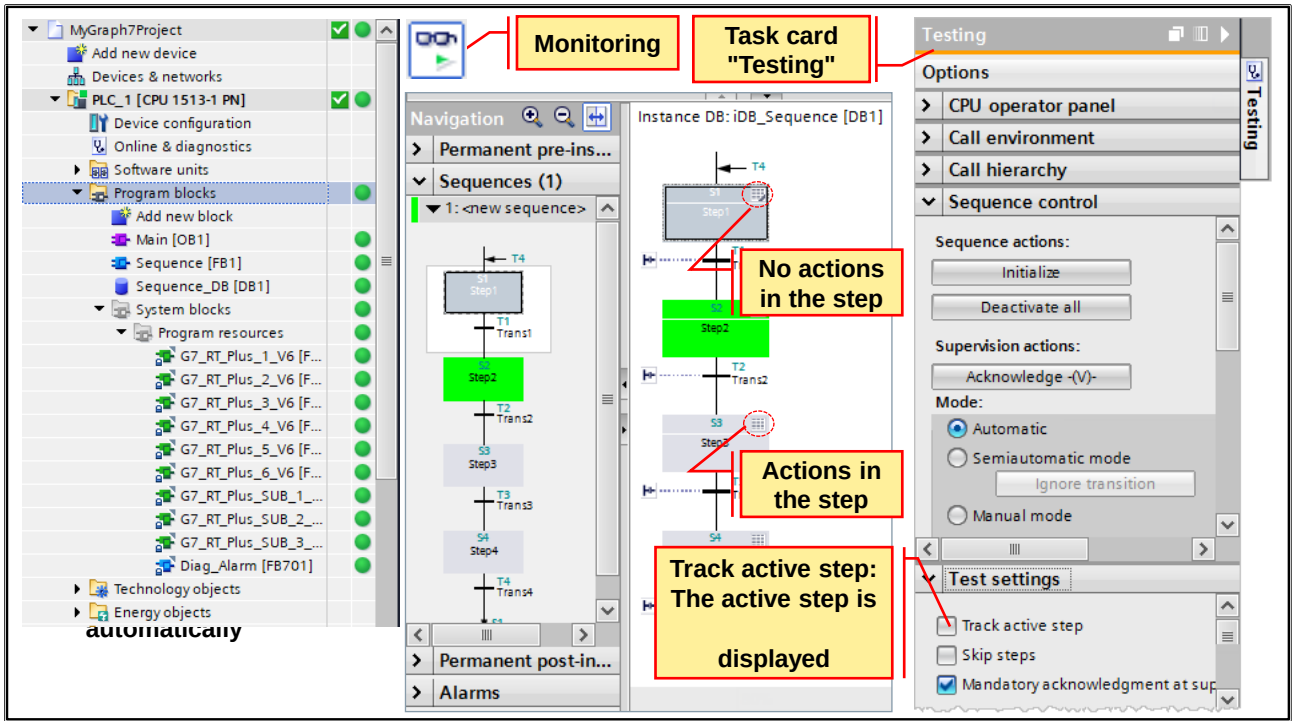
Defining conditions

Conditions in transitions can be configured in the sequence view after expanding the transition



or in the single step view.

3.9. Testing a step sequence



Test functions

The test functions enable a many different test variants in different operating modes, e.g.

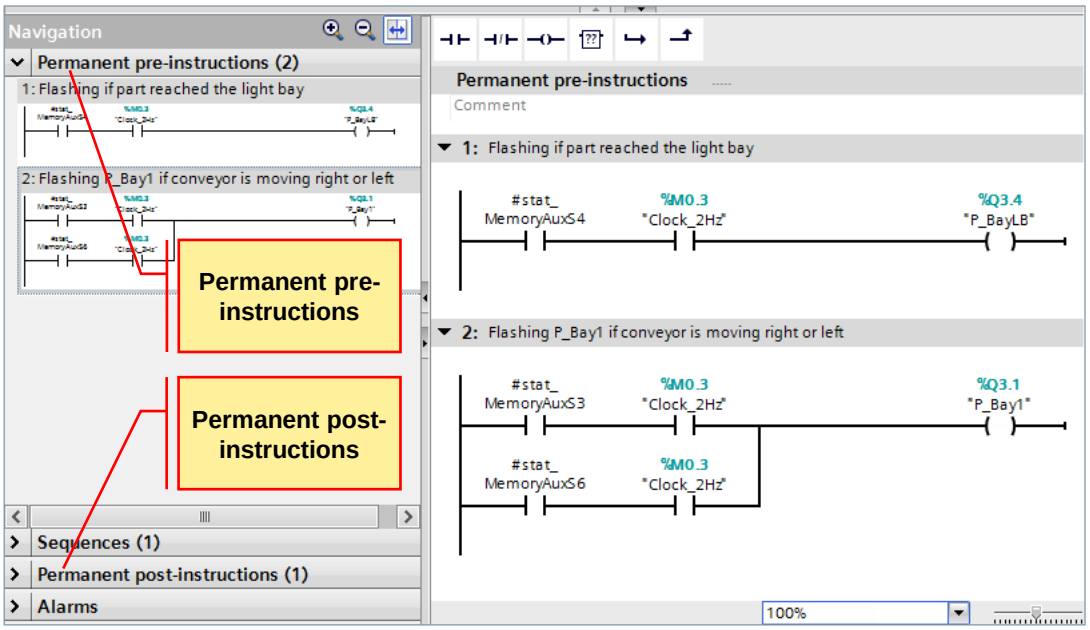
- Display of active steps and steps with faults
- Status display and modify tag
- Switchover between operating modes: Automatic mode, semiautomatic mode and manual mode
- Mandatory acknowledgement with different criteria
- Initialization of the sequence
- Deactivation of all steps
- Etc.

Empty transition, Step advances immediately

Actions are present in the step

Transition contains conditions

3.10. Permanent instructions



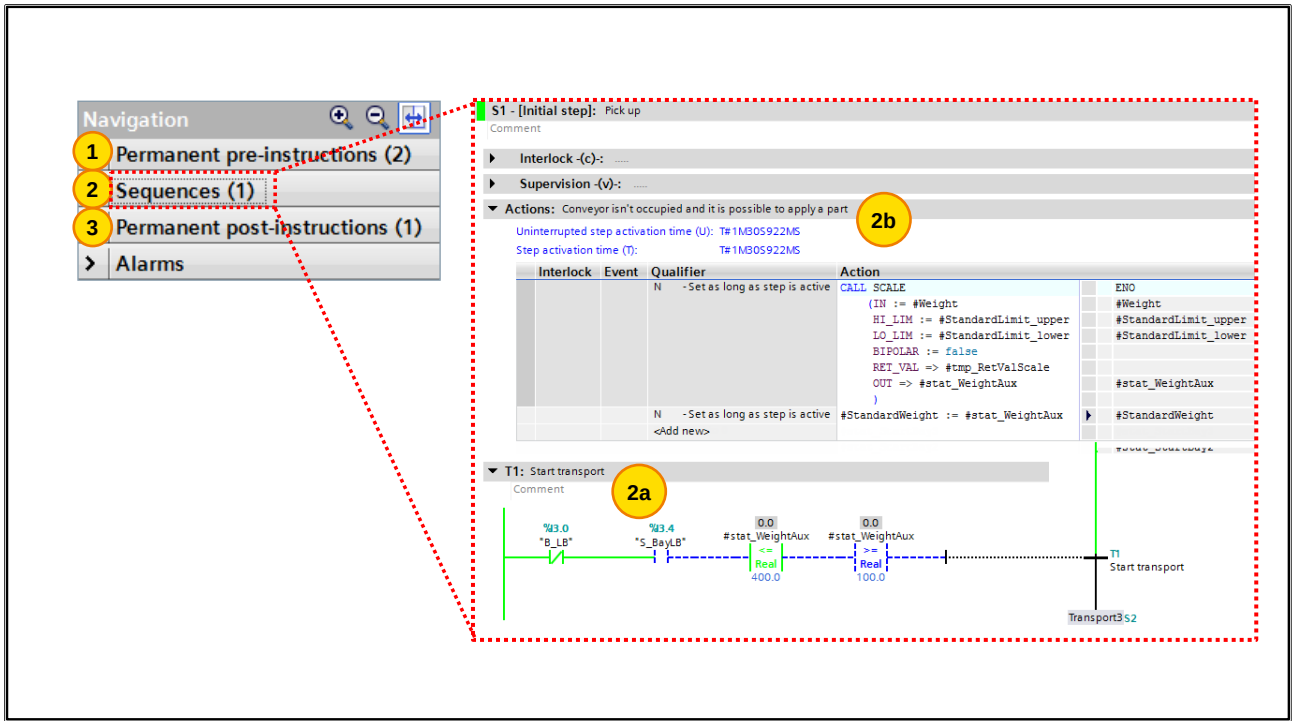
Permanent instructions

Permanent instructions are conditions or block calls that are located before or after the sequencer. These instructions are processed once per block call regardless of the state of the step sequence. The number of instructions before and after the step sequence in an FB is limited to 250 networks. Each instruction is provided with a number. Up to 32 LAD/FBD blocks are allowed within a network.

Conditions that must be true at multiple points in the step sequence can be programmed centrally with permanent instructions.

For example, the flashing of an indicator light in a step can be easily implemented in this way.

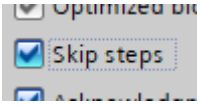
3.11. Processing order of GRAPH FB



Processing order

The processing order of the individual components of a GRAPH FB is made up as follows:

1. Permanent pre-instructions are executed
2. In the sequences, the following steps are carried out:
   a. Checking of transitions
   b. Checking of the interlock and supervision conditions and subsequent execution of the actions of all active steps
   c. (b) does not take place if the "Skip steps" option in the block properties is activated and the preceding and following transitions are fulfilled at the same time.

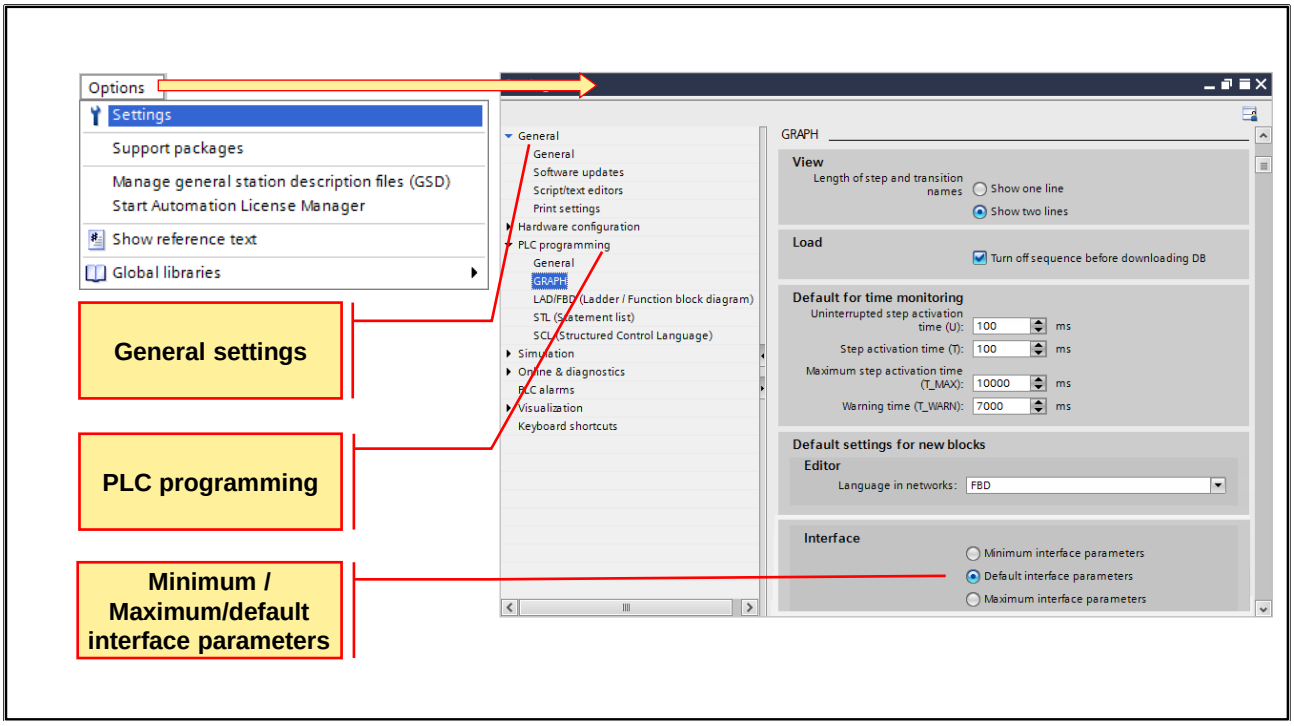


3. Permanent post-instructions are executed

Note

Because the transitions are executed first and then the actions, a temporary tag cannot be used in the steps for the step-enabling conditions in a GRAPH FB.
This is, of course, possible in the pre-instructions!

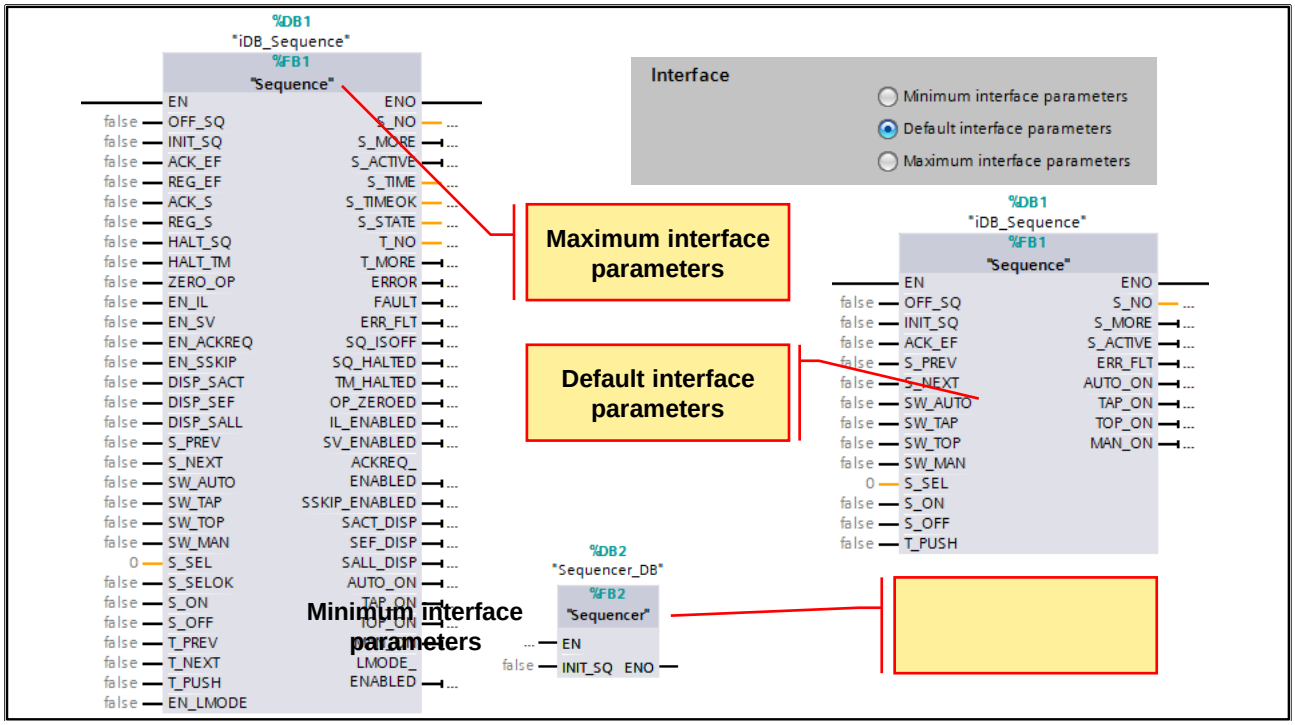
3.12. Settings for the GRAPH editor and compiler



Settings

Besides the general settings for the TIA Portal, there are also GRAPH-specific settings, which can be found in the GRAPH sub-item under PLC programming. Here you will find various default settings as well as the setting for the interface definition when a new GRAPH block is created. The default setting for this is "Maximum interface parameters".

3.13. Call of a GRAPH FB



Sequence FB

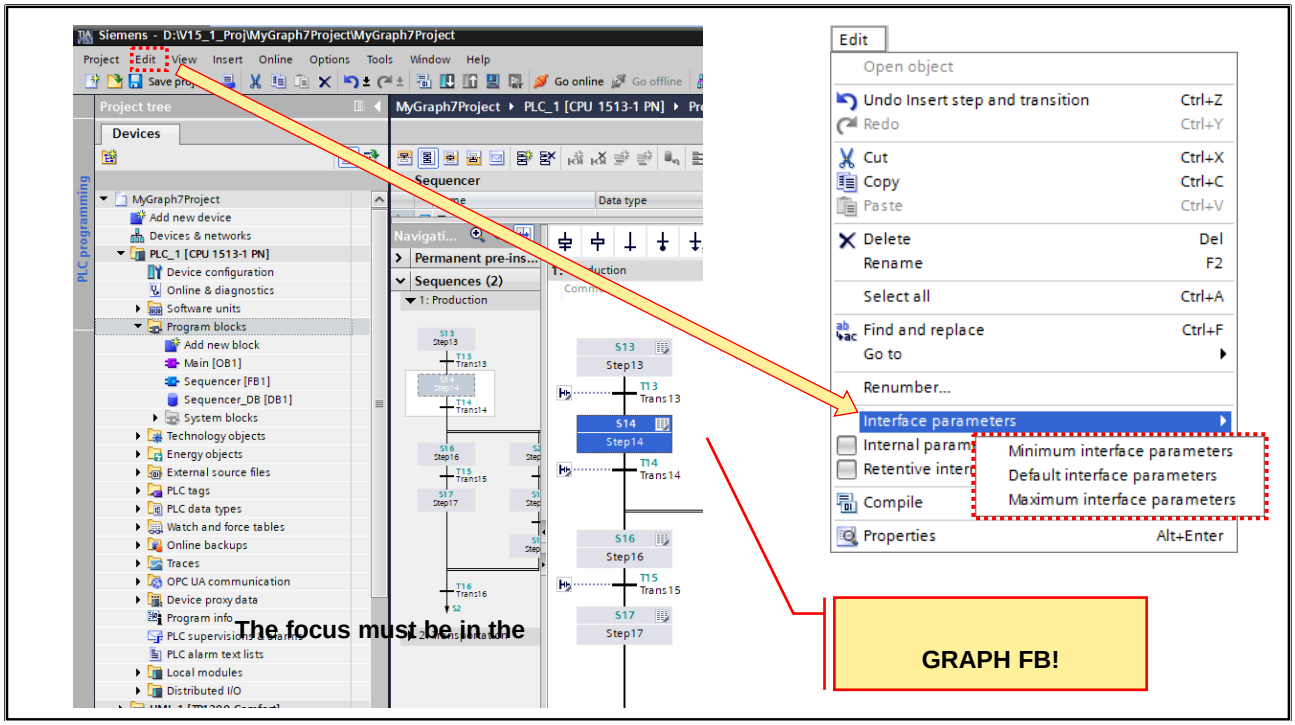
Using the Options → Settings → PLC programming → GRAPH menu command, it can be selected whether the FB is to be initialized with the default or maximum list of parameters when it is created or called up. User-defined parameters are always visible.

The initial step(s) are activated via a positive edge at the INIT_SQ parameter.

This FB also requires an operating mode selection for the call. The default parameter set is always used when the sequencer is to run in various operating modes and feedback about the process and the ability to acknowledge alarms are needed.

The sequence FB always stays at the last activated operating mode in this case. Only through selection of another operating mode can the preceding operating mode be deselected. Unneeded parameters are not initialized.

3.14. Adapting maximum interface parameters subsequently



Maximum interface parameters

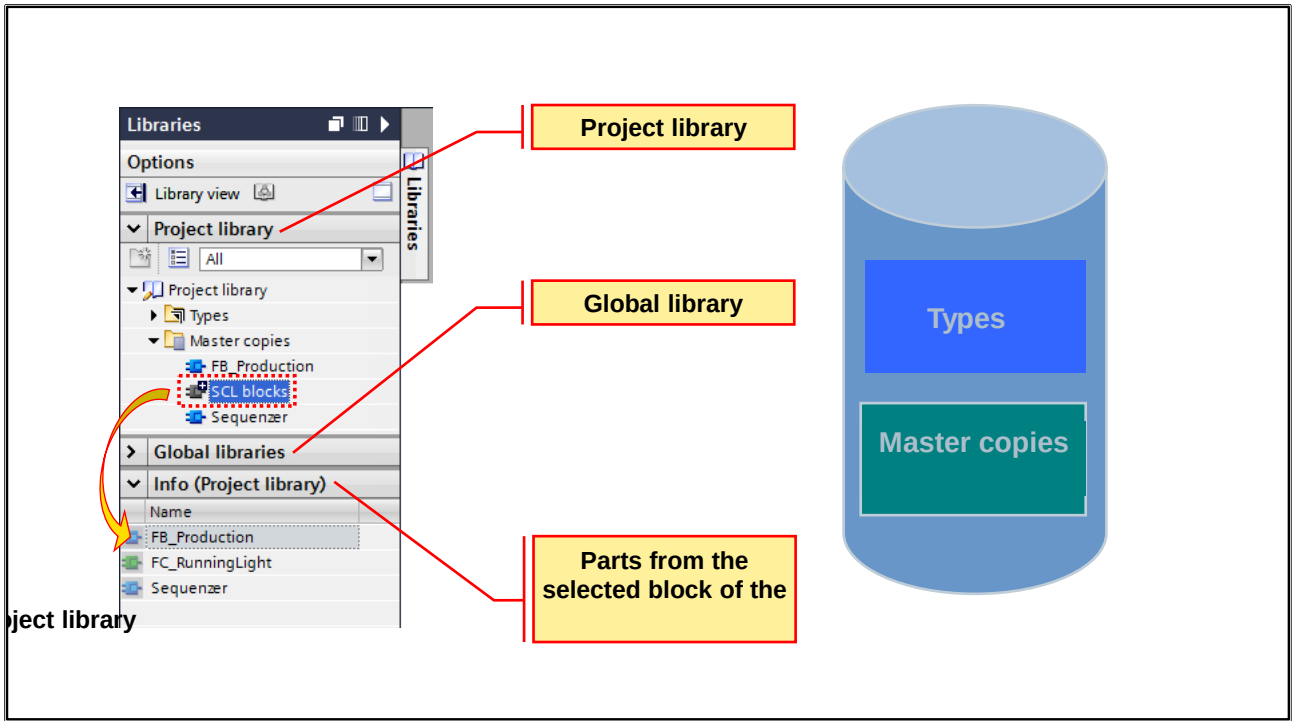
Using the Edit → Maximum interface parameters / Default interface parameters menu command, the user can still change these properties even after the FB has been created.

Note

The focus must be placed in the GRAPH FB so that these interface settings are visible. This is done most easily by pressing the left mouse button on a step or transition.

However, a changeover between the maximum and default interface parameters always requires an update of all calls of this GRAPH FB.

3.15. General information about libraries



Libraries

Generally, project libraries and global libraries are available.

Project library

A project library is characterized by the following:

- One library per project
- Contains objects that are reused only within the project
- Library is opened, saved and closed together with the current project

Global libraries

Global libraries have the following properties:

- Independent of the project
- Can be stored on decentralized drives
- Contains objects that can be reused across projects

Library elements

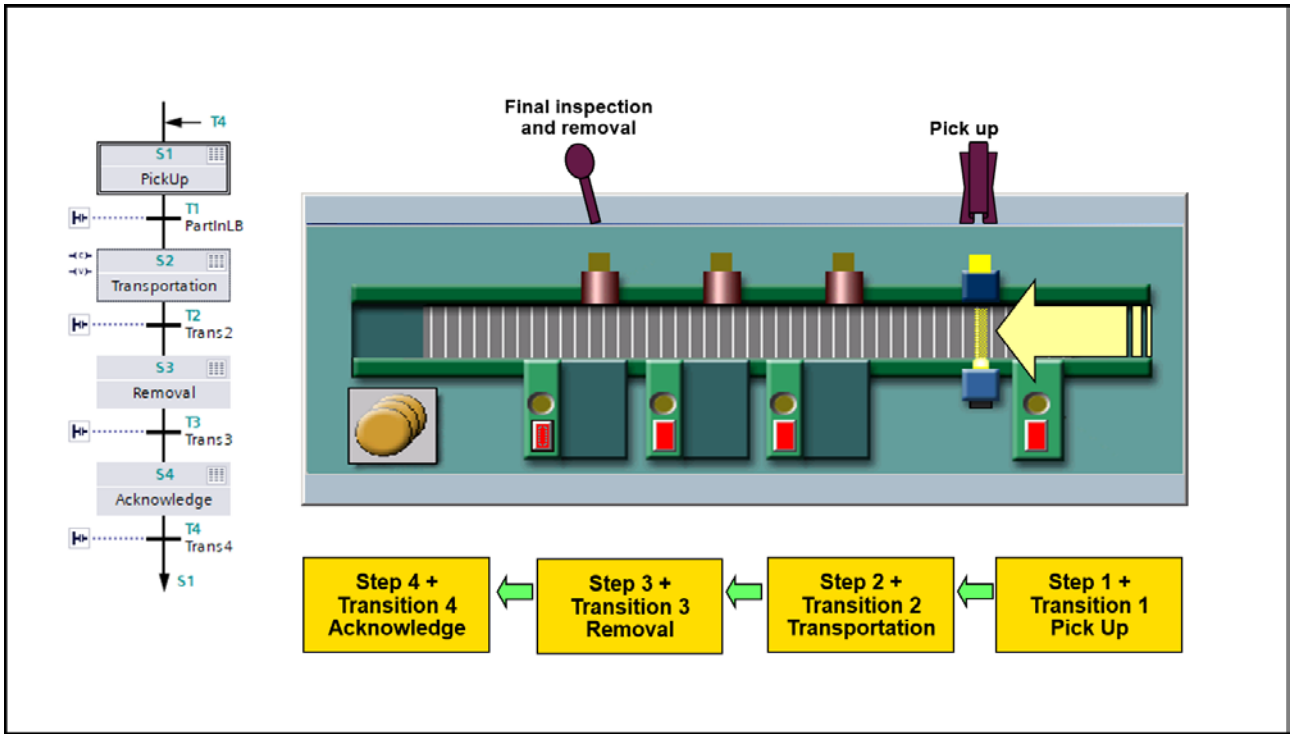
Libraries are structured into types and master copies
Master copies contain any number of copies of blocks, stations, tag lists, etc.
Types contain elements such as data structures and faceplates, which are instantiated by copying them to the work area. The most important criterion is the ability to change instances centrally.

The smallest unit that can be saved in a library is a block.

Note

In contrast to master copies, types can be changed directly in the library.

3.16. Task description



Task

Only a rudimentary step sequence will be programmed to start, but it will later serve as the basis for our manufacturing line as the training course progresses.
For this purpose, create a step sequence that fulfills the following basic functionality.

Procedure

1. **Step 1 – Pick-up / Init:**
   - The indicator light at the light barrier bay "P_BayLB" (%Q3.4) flashes at a frequency of 1 Hz, thus signaling that a blank can be put on the conveyor. The blank must be put through the light barrier for reliable detection.
     When this happens, the LED at the light barrier bay lights up and stays lit. Next, the "S_BayLB" (%I3.4) pushbutton at the light barrier bay can be actuated to advance to the next step.
2. **Step 2 – Transport:**
   - The blank is transported from the light barrier to the proximity switch at bay 1.
3. **Step 3 – Removal**
   - The blank is at bay 1 and can be removed from the conveyor. This is indicated to the user by the flashing of the "P_Bay1" (%Q3.1) LED at 1 Hz.
4. **Step 4 – Acknowledge:**
   - The removal of the blank is acknowledged by pressing the "S_Bay1" (%I3.1) pushbutton. The steady lit "P_Bay1" LED signals that an acknowledgement is possible.
   - If the blank is put back on the conveyor, this causes the "P_Bay1" LED to flash at 1 Hz.

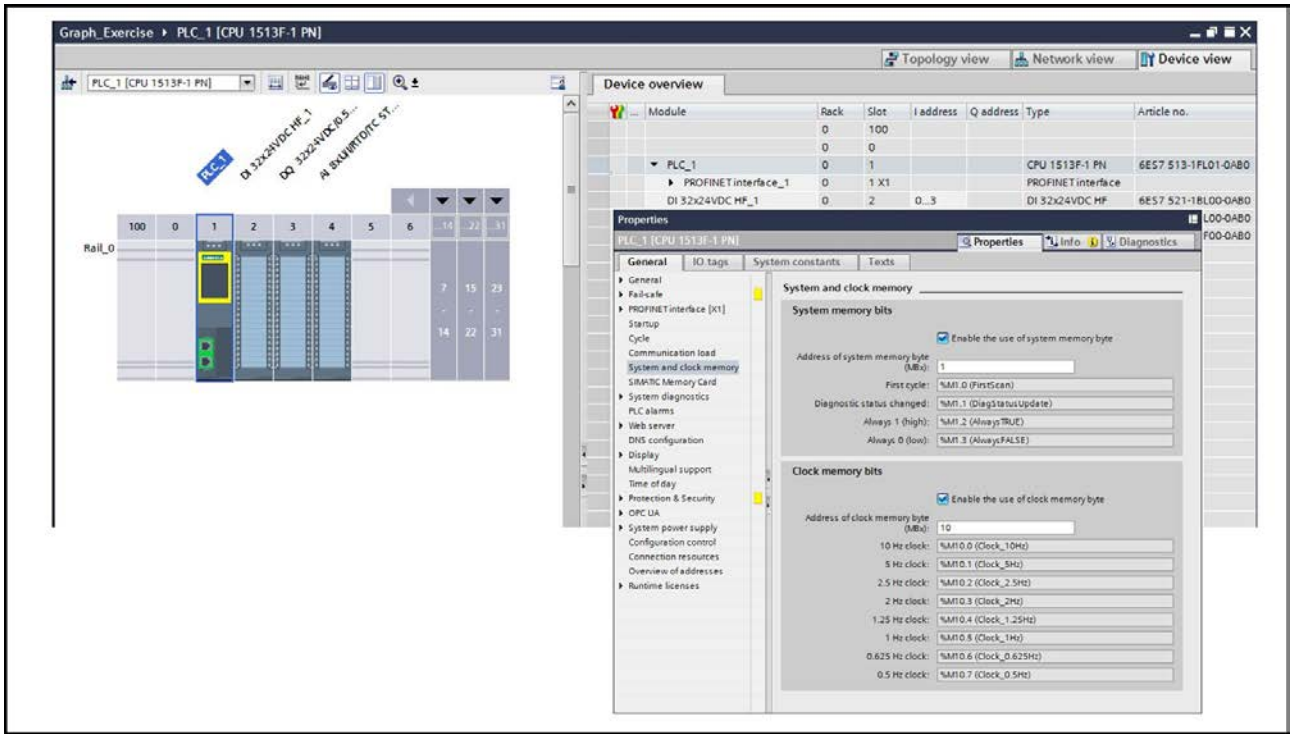
Flashing function

The easiest way to implement the required flashing functions of the "P_BayLB" indicator light in step 1 and "P_Bay1" in steps 3 and 4 is to use permanent instructions. Use temporary tags, if necessary.

PLC tags

Use the preconfigured variable table from the library (Graph_Library_V16.0).

3.17. Exercise 1: Parameter assignment of the S7-1500 CPU



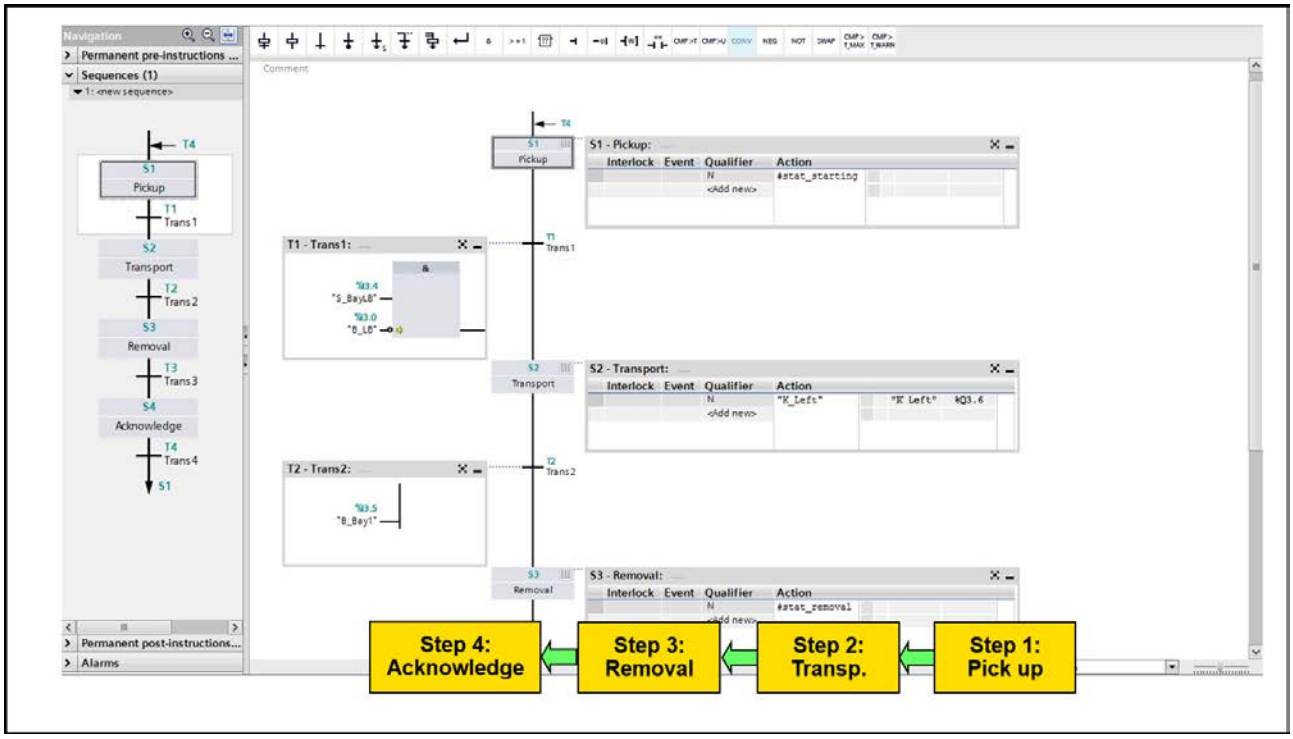
Task

Assign the parameters of the S7-1500 as shown in the slide.

Procedure

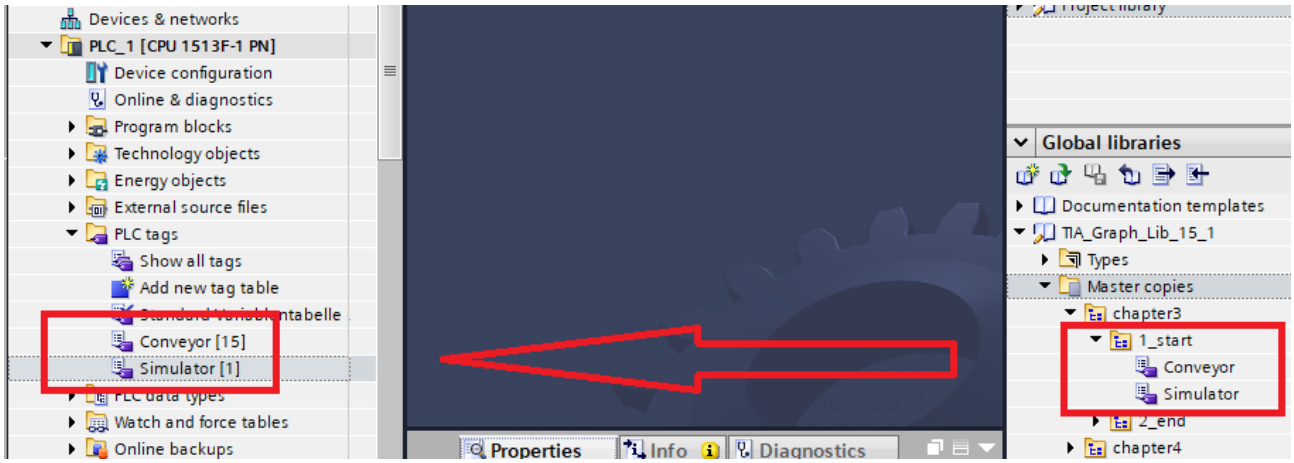
1. Open the "Device configuration" of the PLC you created in chapter 2.
2. Activate use of the clock memory byte with address %MB10.
3. Activate use of the system memory byte with address %MB1.
4. Assign the addresses of the I/O modules as shown in the slide.
5. Save and compile the entire configuration.
6. Download the changed hardware configuration to the controller.

3.18. Exercise 2: Configure a linear sequencer



Task

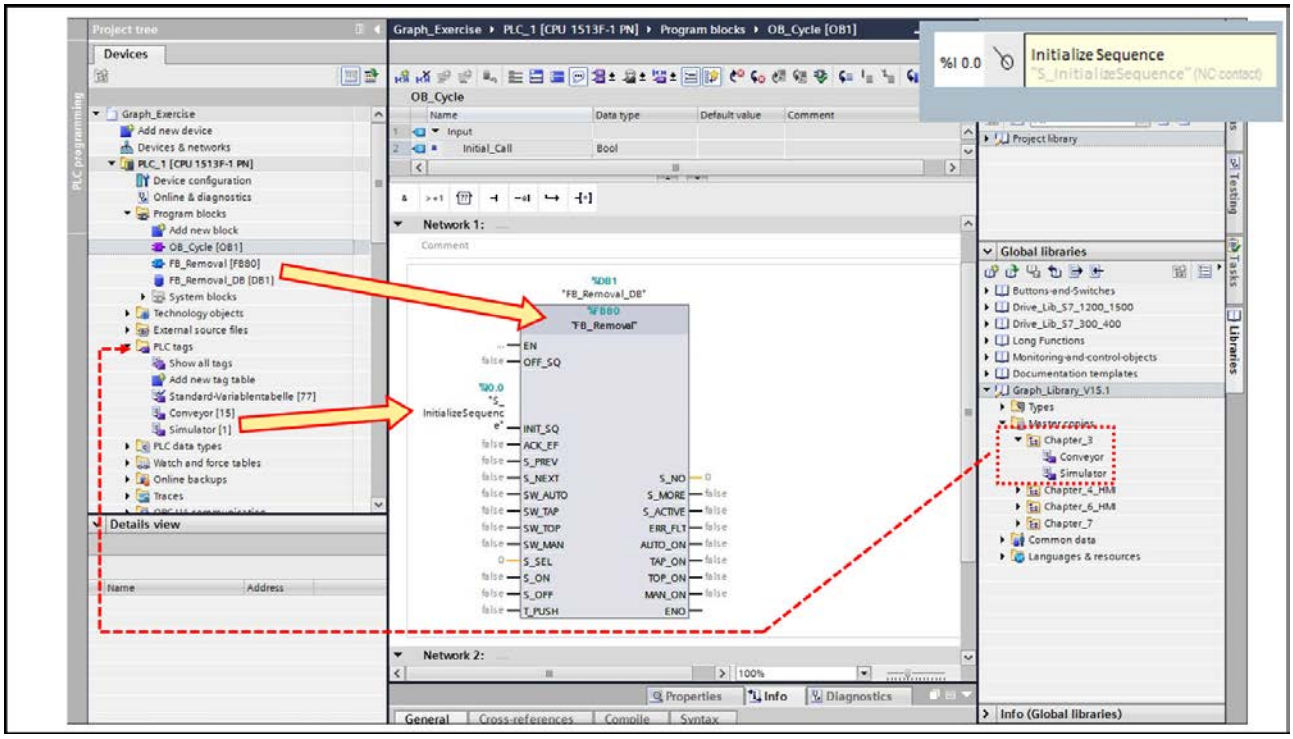
You configure a linear sequencer, assign step and transition names and test the sequence.
You will find two ready-made PLC tag tables in the "TIA_Graph_Lib_16", which is stored under "C:_Archive\TIA-GRAPH*[version]*\".
Drag these into the PLC tags in your project.



Procedure

1. Create a GRAPH FB80 with the symbolic name "FB_Removal".
2. Open the FB80 by double-clicking it and enter the structure of the sequencer according to the task description. Set the work area of the GRAPH editor to the sequence view for this.
3. Configure the necessary steps, actions and permanent instructions. Choose the sequence view or single step view for the work area as you wish. Assign names for the steps and transitions.
4. Save and compile the FB80.

3.19. Exercise 3: Call up of the GRAPH FB in OB1



Task

Call up your created GRAPH FB in OB1 and test it.

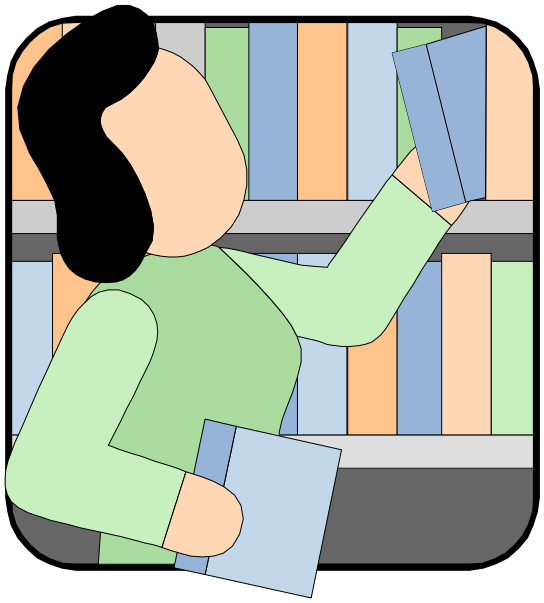
Procedure

1. Call up the FB80 with instance DB80 in OB1.
2. Transfer all blocks to the automation system and test the program with the "Monitor" test function. Alternatively, click the button with the "eyeglasses".
3. Save the completed step sequence block in the local library.

Result

The active step is displayed in "green". If interlock or timeout errors are to be monitored (later exercises), the step involved is displayed in "red" or "orange". The signal states of the conditions and actions can be monitored in the active steps.

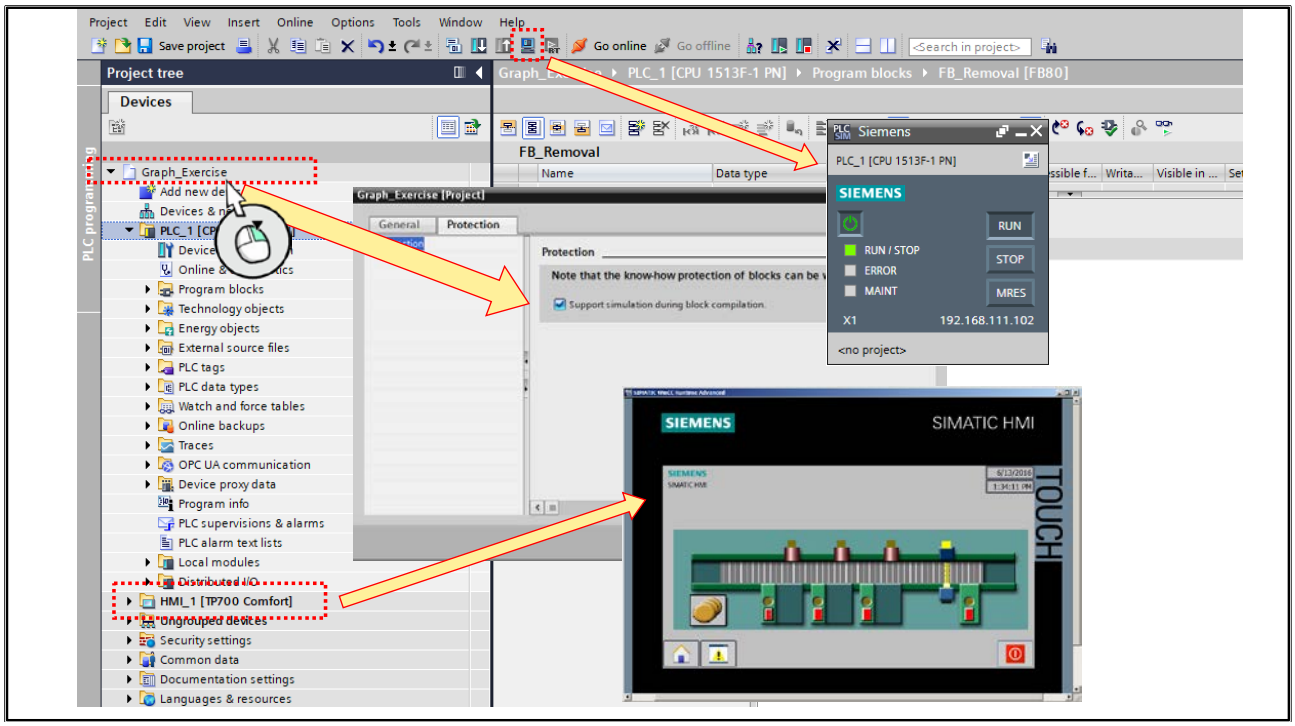
3.20. If you want to know more



Note

The following pages provide additional information or serve as a reference that can be consulted for more complete information on a subject.

3.1.1. Simulation of PLC and HMI devices



Starting the simulation:

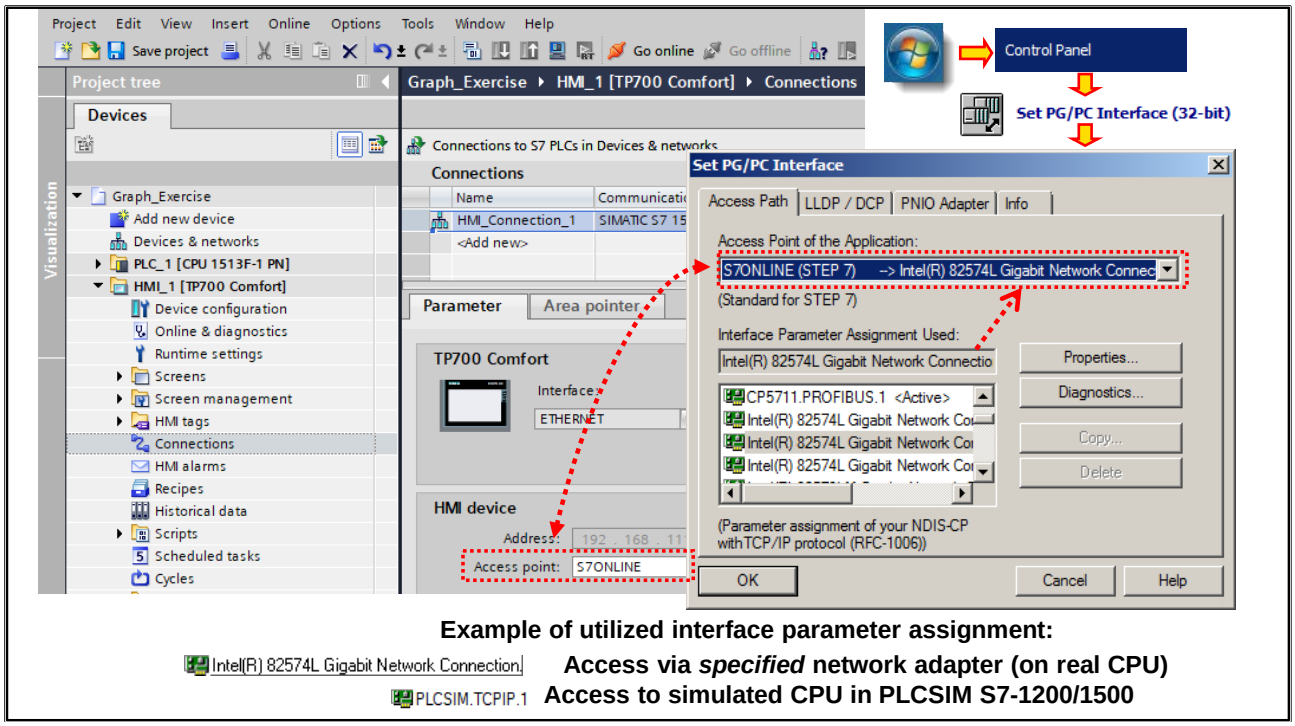
The simulation of a PLC or HMI device is started in the TIA Portal using the "Start simulation" button. The desired device, PLC or panel must be selected before the simulation.

Note

The simulation of an HMI device can be active in parallel with a connection to a real device or PLC.
By contrast, the simulation of a PLC deactivates all other interfaces and allows only communication via the PLCSIM interface PLCSIM. Communication with the simulation of an HMI device is always possible, however!

Before you can simulate PLC programs with S7 1500/1200, the corresponding compiler option must be activated in the properties of the project root. "Support simulation during block compilation". In particular, if know-how protected block are to be simulated, the simulation option must be activated before the block protection is activated. However, this also reduces the quality of the know-how protection.

3.1.2. Setting the S7 online interface for the simulation



Example of utilized interface parameter assignment:

Intel(R) 82574L Gigabit Network Connection **Access via *specified* network adapter (on real CPU)**

PLCSIM.TCPIP.1 **Access to simulated CPU in PLCSIM S7-1200/1500**

Settings

If the simulation of the HMI device is to be able to communicate with a real or simulated PLC, another setting must be made:
the interface via which this communication is to take place must be assigned.

To do so, the right access point must be specified in the connections of the HMI device (default setting is "S7ONLINE") and the right PG/PC interface must, in turn, be assigned to this (see "Set PG/PC Interface" in figure).

The option exists to select a real network adapter if communication is to be established with a controller connected using this network adapter (selection of the corresponding network adapter). In addition, it can be specified that communication will be with a simulated CPU. The PLCSIM interface is then selected for a classic PLC and the PLCSIM S7-1200/S7-1500 interface for an innovated controller.

Table of contents

4

4. Interlock, Supervision and Alarms 4-2

4.1. Interlock and Supervision 4-3
4.2. Interlock and Supervision: Additional functions in the step 4-4
4.3. Interlock as additional interlock condition 4-5
4.4. Use of the interlock in transitions 4-7
4.5. Interlock in the online view 4-8
4.6. Use of interlock in actions 4-9
4.7. Supervision (..... 4-10
toring)
4.8. Supervision in the online view 4-12
4.9. Supervision error with resulting interlock condition 4-13
4.10. Supervision error with resulting interlock condition 4-15
4.11. Task description 4-16
4.12. Exercise 1: Create a new GRAPH FB 4-17
4.13. Exercise 2: Expand the conveyor functionality 4-19
4.14. Exercise 3: Interlock and Supervision 4-20
4.15. Configuring alarms 4-21
4.15.1. Adding step-specific alarm texts 4-22
4.16. PLC alarms for displaying sequence errors 4-23
4.17. Display and acknowledgement 4-24
4.18. Activating the web server on the CPU 4-25
4.19. Exercise 4: Display and acknowledge PLC alarms 4-26
4.20. Exercise 5: Display of sequence errors in the web server 4-27
4.21. Creating your own alarm class for the project 4-29
4.22. Displaying alarms on the HMI device 4-30
4.23. Alarm components and properties 4-32
4.24. Steps for alarm configuration 4-33
4.25. Alarm classes 4-34
4.26. Alarm settings 4-35
4.27. Exercise 6: Create touch panel 4-36
4.28. Exercise 7: Copying elements from the library 4-40
4.29. Exercise 8: Configure the alarm view 4-42
4.30. Exercise 9: HMI Runtime settings 4-43

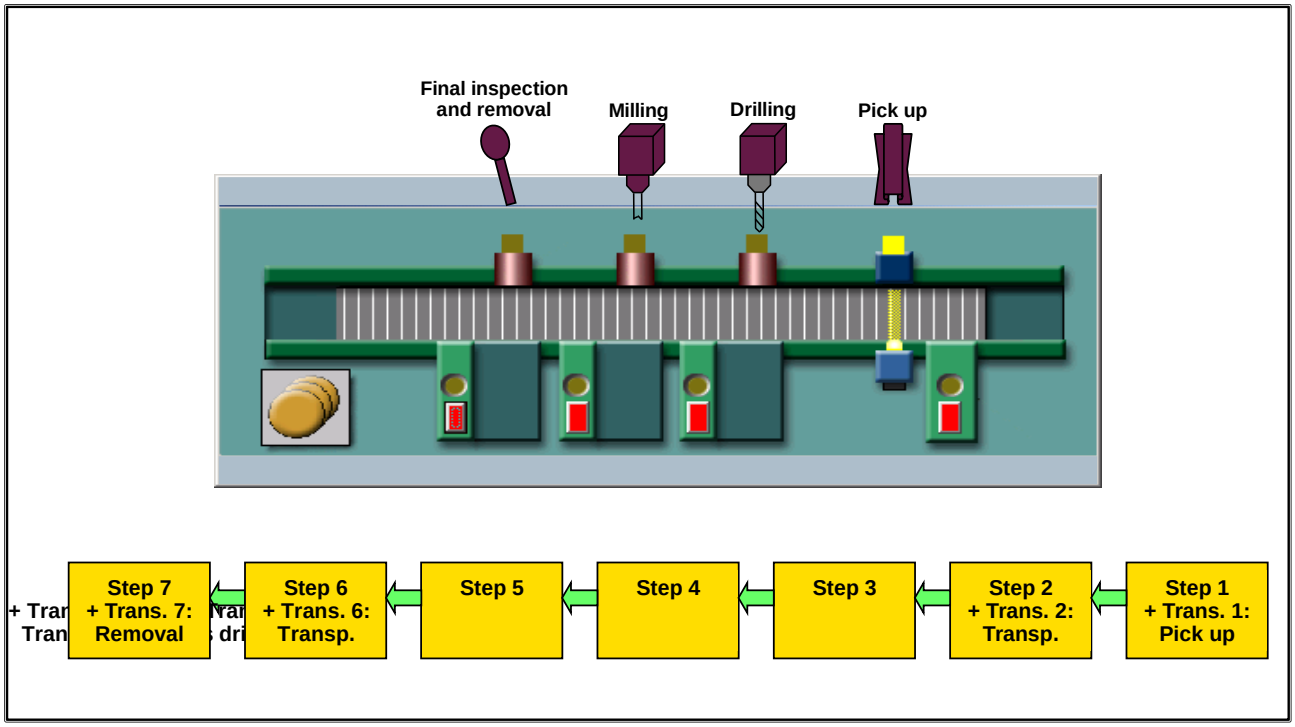
4. Interlock, Supervision and Alarms

The participant should ...

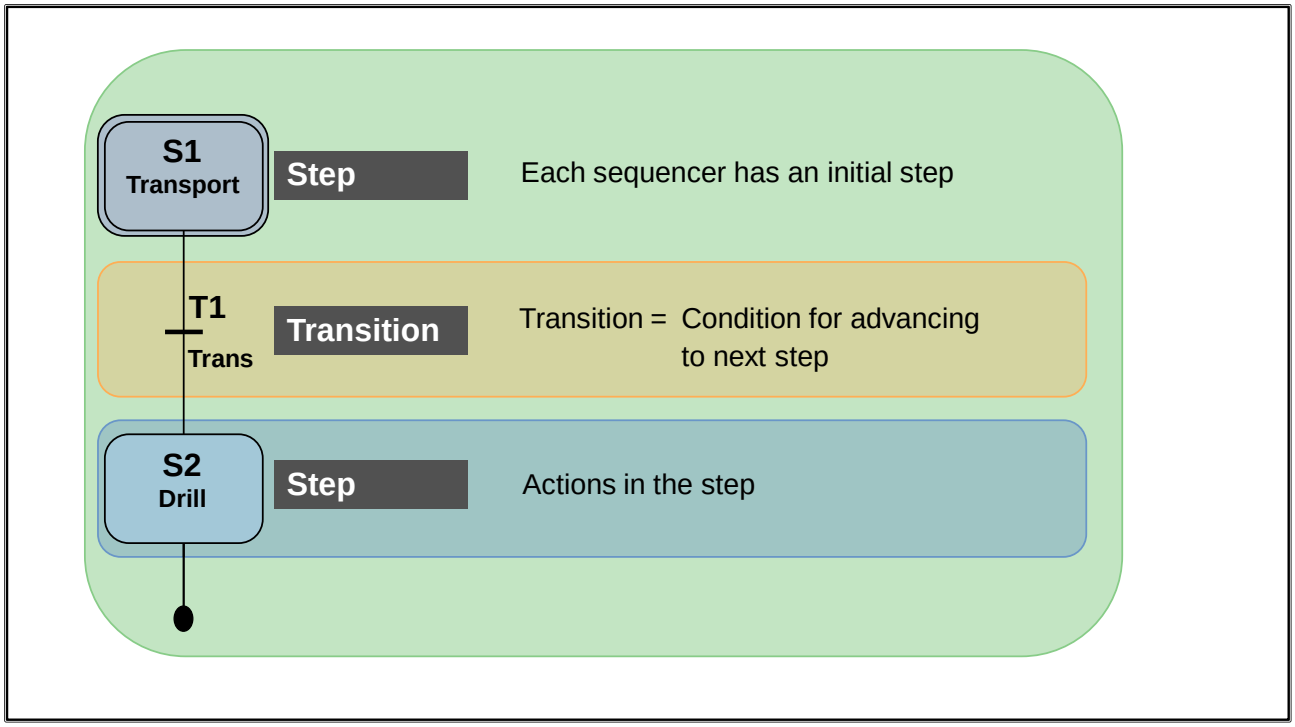
… be able to understand and use the additional interlock and supervision of a step sequence

... be able to display and understand the error messages of a sequencer in the form of CPU alarms, in the HMI alarm system and on the CPU web server

4.1. Interlock and Supervision



4.2. Interlock and Supervision: Additional functions in the step



Interlock

- An action marked with "C" (Condition) is only executed when the interlock condition (Interlock) of the step is true.

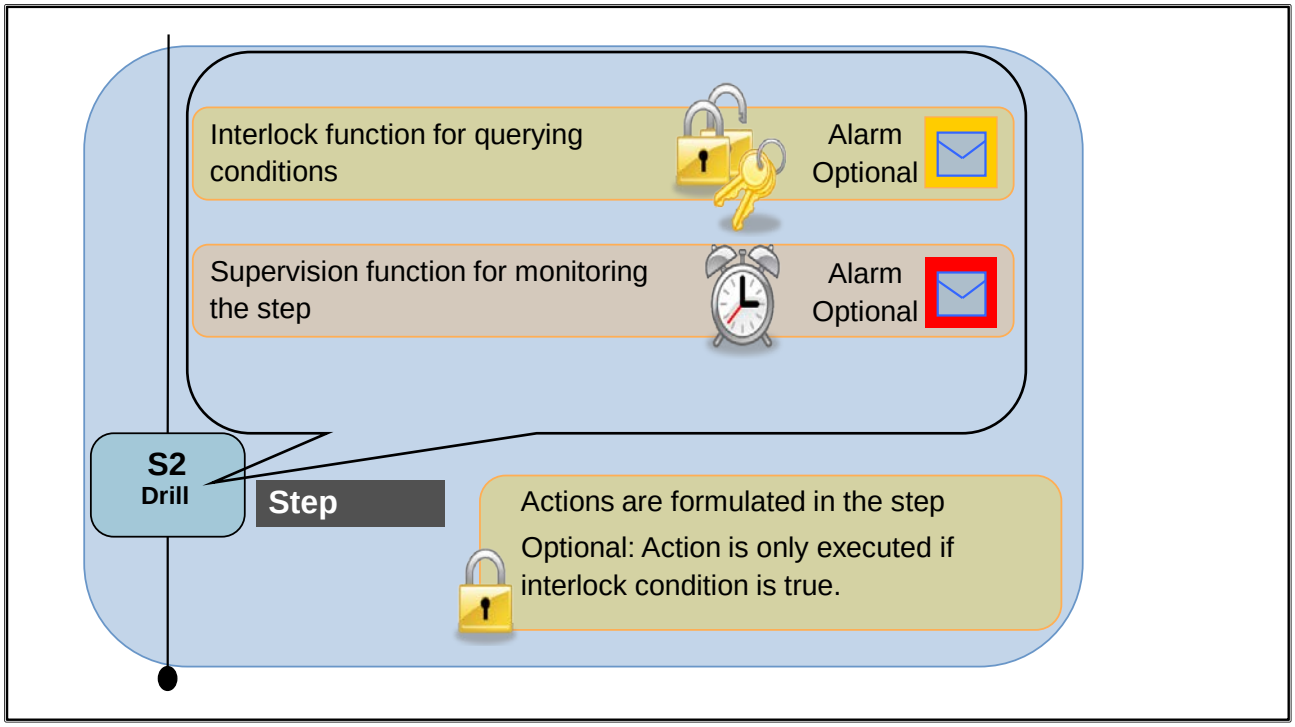
Interlock	Event	Qualifier		Action		
-(C)-		N	- Set as long as step is active	#tmp_drillActive		
		N	- Set as long as step is active	"P_Fault"	"P_Fault"	%Q0.7
-(C)-		N	- Set as long as step is active	"P_Interlock"	"P_Interlock"	%Q0.6
		<Add new>				

- An interlock error is present when the condition is false ("C" = 0). The actions marked with C are then not executed. The step is marked in orange and an interlock error is generated.
- An interlock error does not prevent an advancement to the next step when the transition condition is true!

Supervision

- A supervision function triggers an alarm based on a user-programmed condition.
- However, a supervision does not prevent execution of the actions in a step.
- In contrast to an interlock, an active supervision function automatically blocks the advancement to the next step even if the transition condition is true.

4.3. Interlock as additional interlock condition



Interlock

Actions in the step can be made dependent on conditions. Such step interlocks are marked by the "C" symbol.

When the interlock condition is true, all actions that are logically combined with the "C" interlock in the active step are executed. If the result of the logic operation of the interlock conditions is false, a fault is present.

Fault

Under the fault condition "Interlock is not satisfied", the following rules apply:

- Actions that are logically combined with the interlock are **not** executed.
- An interlock error "L1" is signaled for one cycle.
- The uninterrupted step activation time "Step_x.U" stops.
- The step activation time "Step_x.T" continues running.
- Advancement to next step when transition condition is true.

to TRUE.• Setting of the FB parameter ERR_FLT

Programming

The interlock is programmed in the "Single step" view. Bit logic and comparison operations are permitted. A programmed interlock is represented in all display types of the sequence with a "C" at the top left corner of the step.

Empty interlock

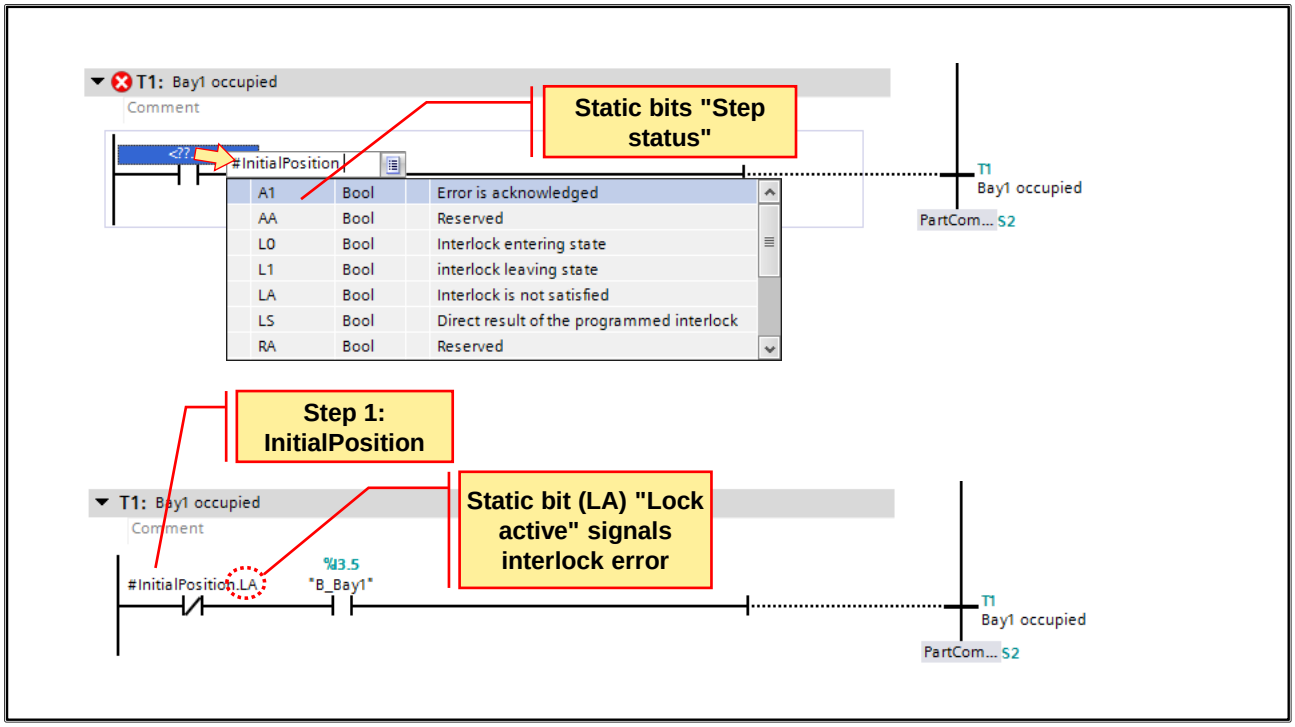
An empty interlock without LAD/FBD elements behaves like a true interlock condition

Note

The FB parameter ERR_FLT = 1 signals a missing interlock condition;
ERR_FLT = 0 means no fault is present.

While the uninterrupted step activation time "Step_x.U" stops when an interlock error is present, it resumes running again starting from that time once the error status changes to outgoing!

4.4. Use of the interlock in transitions



Interlock query

All incoming, outgoing and pending interlock events can be evaluated in the program.

For example, a query of the static bit "Step_x.LA" ("Lock active") can be used to prevent the subsequent transition from advancing to the next step under fault conditions even though proximity switch 1 (%I3.5 on the slide) is occupied.

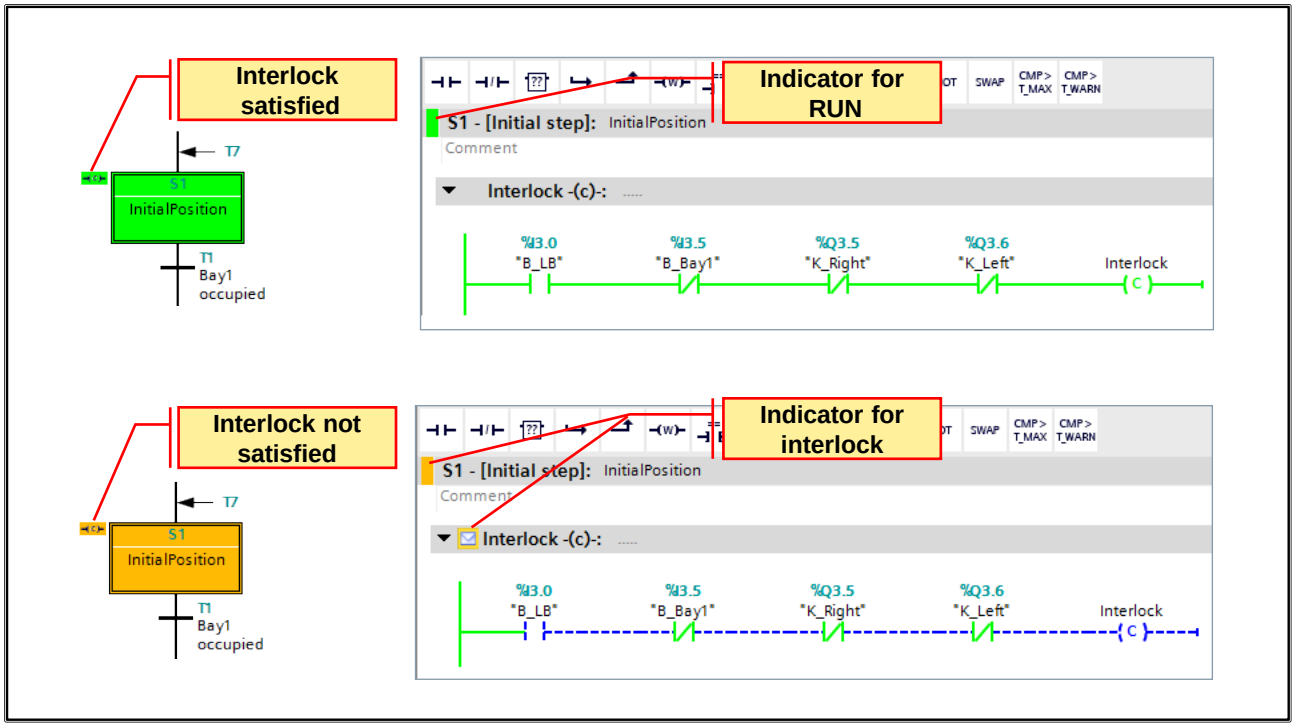
This situation is marked in the slide for an active interlock in the Initial position step:

- #InitialPosition.LA,

i.e. an advancement to the next step is not possible when an interlock fault is present.

An incoming interlock error is marked by the static bit "Step_x.L1", an outgoing interlock error by the static bit "Step_x.L0".
These are comparable to an edge evaluation because they exist only for exactly one cycle!

4.5. Interlock in the online view



Interlock in the online view

The left side of the slide shows the true interlock condition with the following conditions:

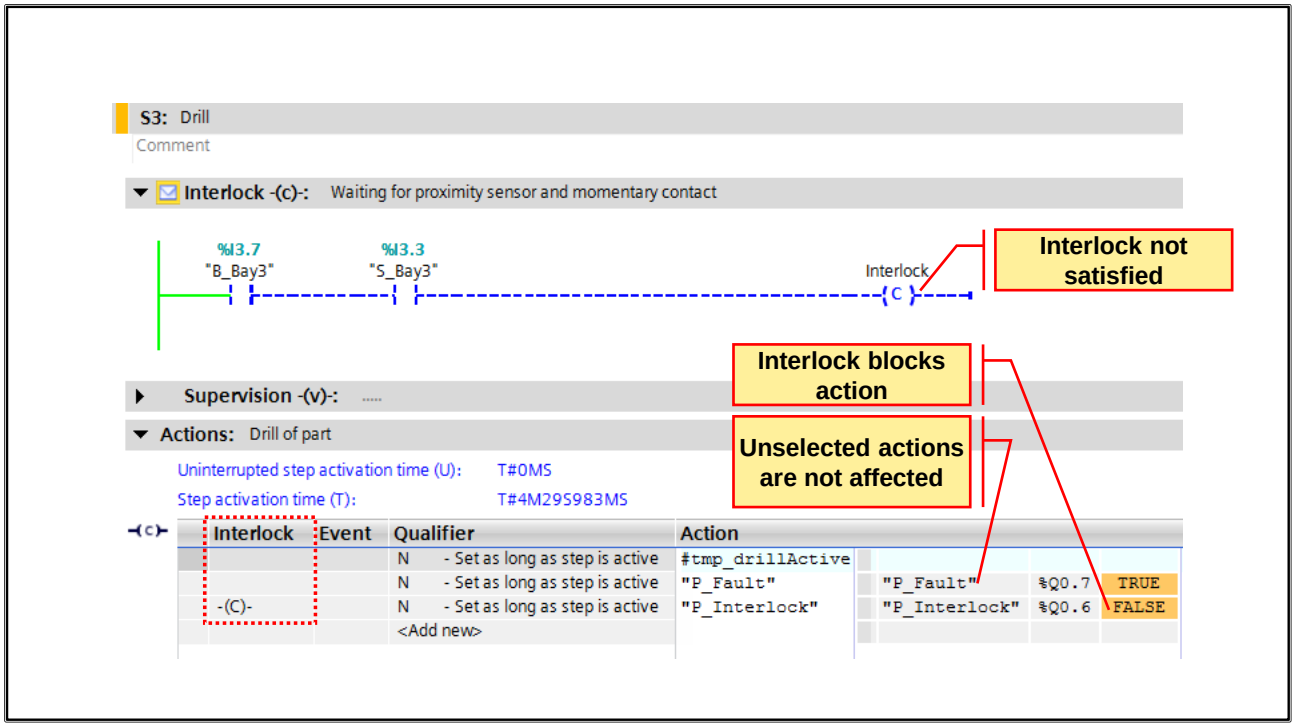
- Light barrier is free
- The bay 1 proximity switch is not occupied
- The conveyor motor is not running to the left or right

In this case, C=1 in the "InitialPosition" step and the actions logically combined with C are executed.

For the case C=0, which is shown on the right side of the slide, the interlock conditions are not true, because a workpiece is blocking the light barrier, for example.

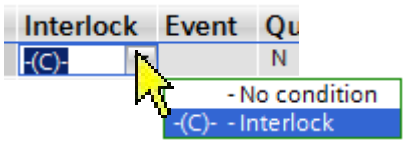
In this case, the actions are not executed and the subsequent transition is disabled using the logic operation with static sequence bit "Lock active" (LA).

4.6. Use of interlock in actions



Interlock usage

An interlock can be used in order to execute actions only when a precondition is true. For this purpose, the interlock query must be enabled for each action that is to be "interlocked" in this way.

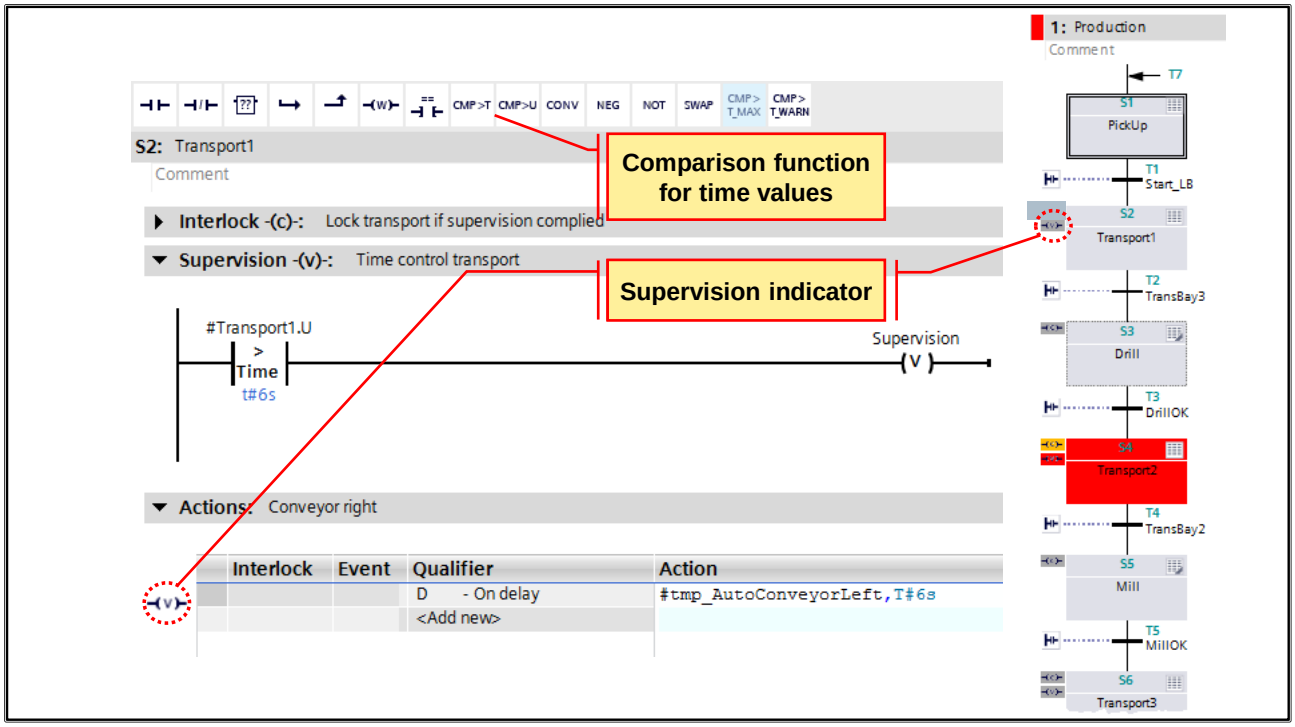


If the interlock condition was true but is no longer true at the next jump into the step sequence, the associated actions are also no longer executed automatically.

An interlock condition that is false does not block the advancement to the next step, however. To prevent this advancement, the interlock must be used in the transition.

Actions not marked with "C" such as the display of a fault using the "%Q0.7" output are not affected.

4.7. Supervision (step monitoring)



Supervision

The supervision function is a step monitoring condition. If the result of logic operation of the supervision condition in the active step is true, a fault is present. As long as the activation time minus any malfunction periods of step 3 "Conveyor Right" is less than 6 s, the RLO = 0. As soon as the activation time exceeds 6 s, the RLO changes to 1 and a supervision error is present. The following rules apply to this:

- No advancement to the next step despite the transition condition being true.
- Setting of the FB parameter ERR_FLT to TRUE.
- The current step stays active, all actions continue to be executed.
- A supervision error "V1" is signaled for one cycle.
- The uninterrupted step activation time "Step_x.U" stops.
- The step activation time "Step_x.T" continues running.

Programming

The supervision is programmed in the "Single step" view. Bit logic and comparison operations are permitted. A programmed supervision is represented in all display types of the sequence with a "V" at the top left corner of the step.

The static bit "Step_x.VA" signals that a supervision error is present "VA" means "Supervision active".

Empty supervision

An empty supervision without LAD/FBD elements is always OK, i.e. ERR_FLT = 0 means no supervision fault.

Note

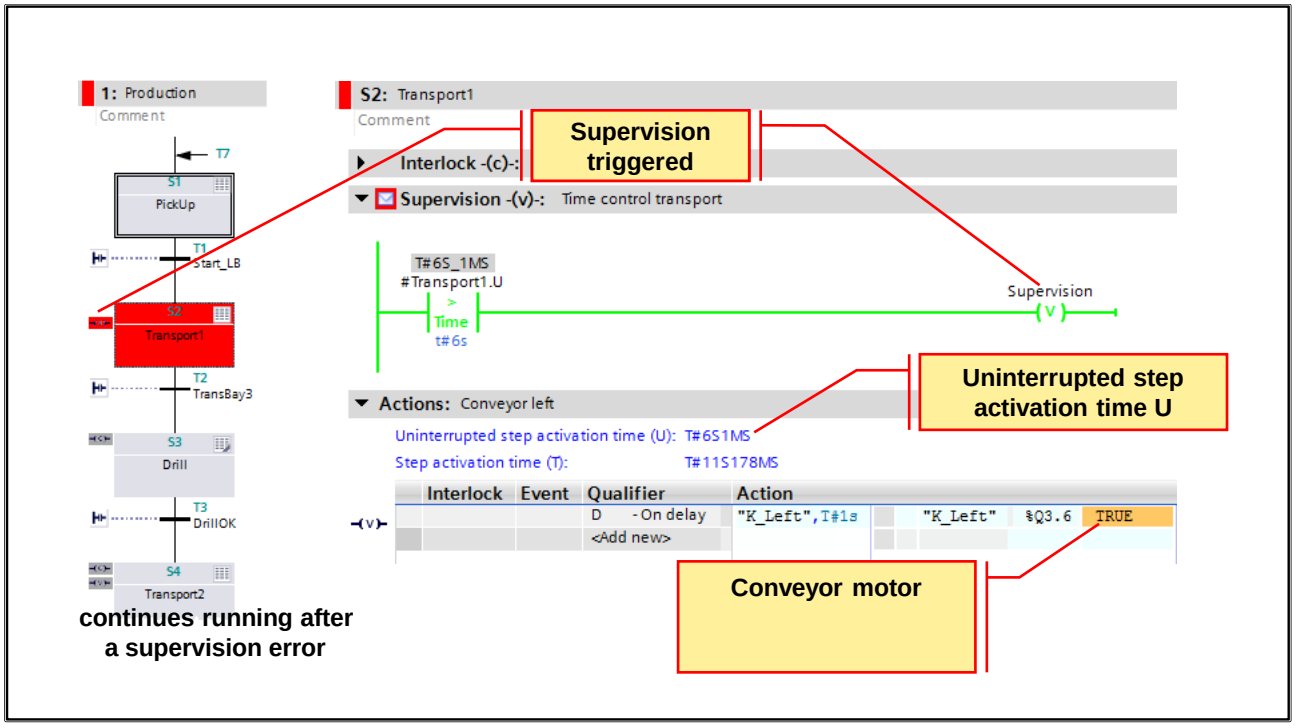
All sequence data can generally be called up (preferably symbolically) at any time within the sequential control using the instance DB.

By programming additional event queries, many different evaluation and interlocking mechanisms are possible with the interlock and supervision functions. This will be demonstrated in further examples.

In the case of an incoming supervision error, the uninterrupted step activa
is stopped. In contrast to an interlock error, however, it restarts at 0 s after the error is acknowledged.

A supervision error must always be acknowledged!

4.8. Supervision in the online view



Supervision in the online view

The left side of the slide shows the triggered supervision with the condition:

- The uninterrupted step activation time "U" must be greater than T#6s.

The step sequence is processed without errors without triggering the time monitoring.

At a triggered supervision, "Transport1.V1" becomes TRUE for one cycle and can thus be queried for the occurrence of a supervision error. As long as the supervision error is present, this can also be displayed by the static bit "Transport1.VA" (supervision active).

At a triggered supervision fault, the conveyor motor continues running even if the subsequent transition has triggered. This is because the advancement to the next step is prevented by an active supervision.

If the supervision error is then acknowledged, the "Transport1.V0" bit is set for one cycle.

Note

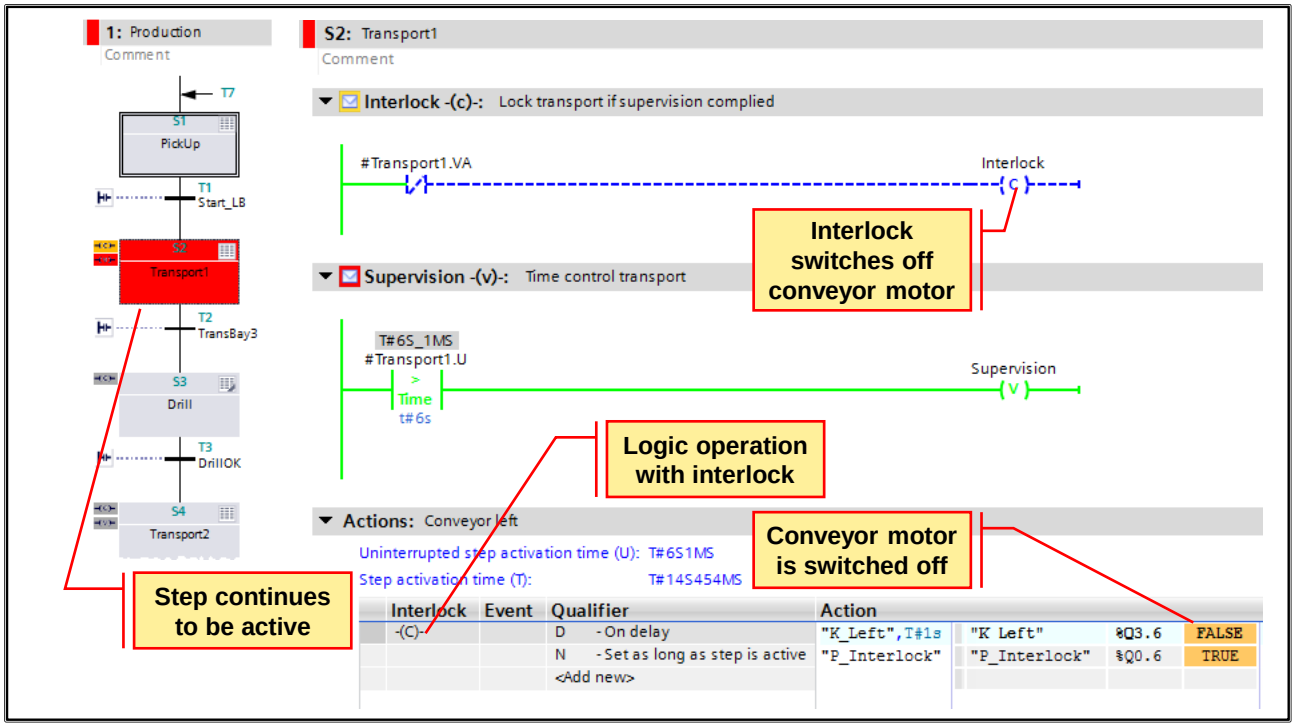
Switching the CPU to Stop state and back to RUN mode does not cause the conveyor motor to switch off. Only the initialization of the sequence using the FB input parameter INIT_SQ or using the sequence control in test mode remains as an emergency measure.

The switching off of the conveyor motor must therefore be solved in a different way.

Time values

In many processes it is necessary to monitor the duration of an operation minus any malfunction periods. The necessary duration of a stirring process, for example, can be 10 milliseconds, irrespective of how long the respective step was disrupted. To perform a time comparison, the uninterrupted step activation time "StepName.U" is used.

4.9. Supervision error with resulting interlock condition



Example

Step S3 "Conveyor_Right" is time-monitored in the example. If the uninterrupted step activation time exceeds the duration of 6 s because the step enabling condition for the next step is not true during this time span, the sequence with a supervision error goes to fault state.

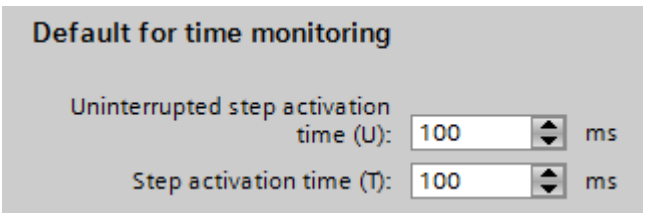
In spite of the stopped sequence, however, step S3 remains active and the action continues to be executed.

To bring the plant to a safe state, the query "Monitoring error occurred in step S3" is evaluated in the step interlock using the inverted bit "#Transport1.VA", and the output "K_Right" is programmed dependent on the interlock.

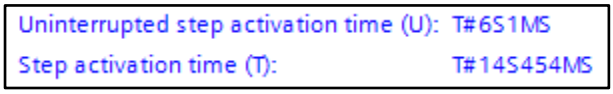
If the supervision error occurs, the interlock condition "C" goes to zero signal (C=0) and the output "K_Right" is disabled.

Time values

The default values for the time monitoring of the various step activation times are stored in the settings and can be adapted there as needed:



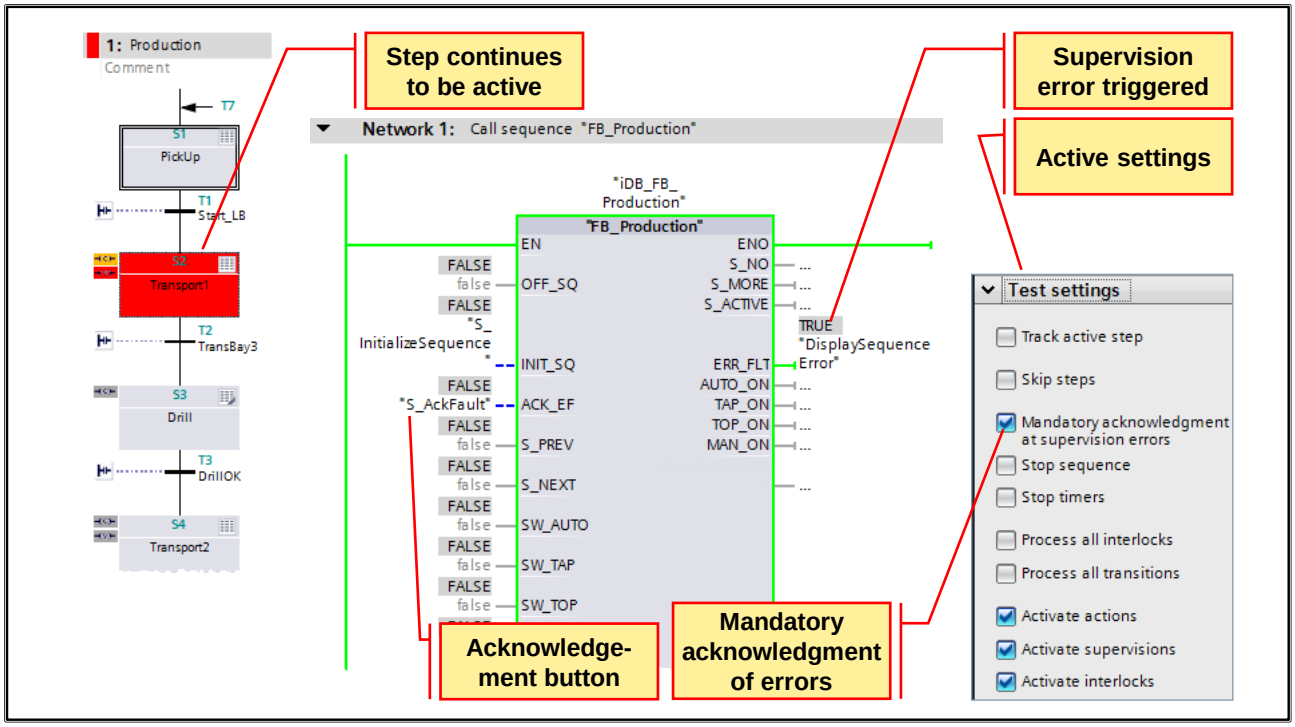
- Step_x.T monitors whether the total activation time of step x is less than the specified value.
- Step_x.U monitors whether the activation time minus the step interlock time in step x, thus the uninterrupted step activation time, is less than the specified value.



Note

The uninterrupted step activation time "U" is reset to the value "T#0s" following acknowledgment of the supervision. The step activation time "T" continues running after error acknowledgment!

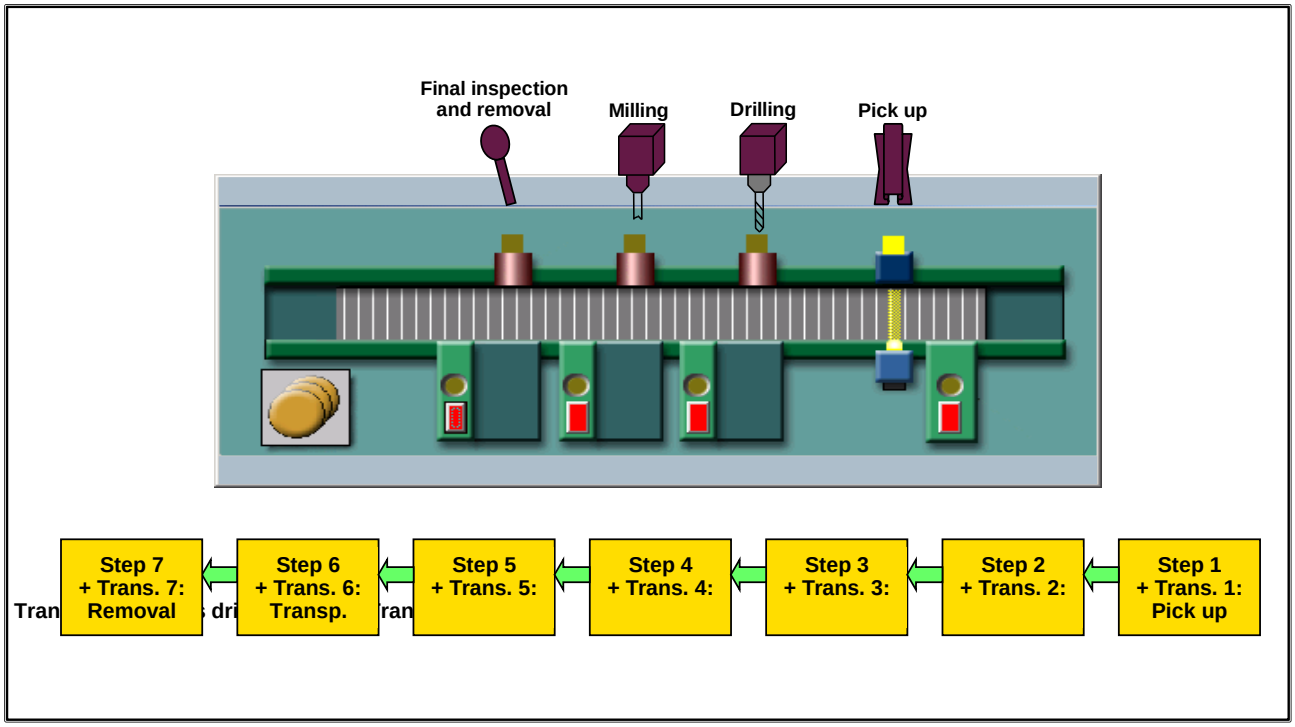
4.10. Supervision error with resulting interlock condition



Example

By means of the test setting "Mandatory acknowledgment of supervision errors", the supervision error can be acknowledged using the "ACK_EF" parameter of the GRAPH FB. In the previous example, the conveyor motor continues running until the subsequent transition is triggered (part in light barrier or workpiece reaches proximity switch 1 on the return). If this is not the case within 6 s, the step triggers another supervision error.

4.11. Task description


Final inspection and removal
Milling
Drilling
Pick up
Tran
Step 7 + Trans. 7: Removal
Step 6 + Trans. 6: Transp.
Step 5 + Trans. 5:
Step 4 + Trans. 4:
Step 3 + Trans. 3:
Step 2 + Trans. 2:
Step 1 + Trans. 1: Pick up

Task

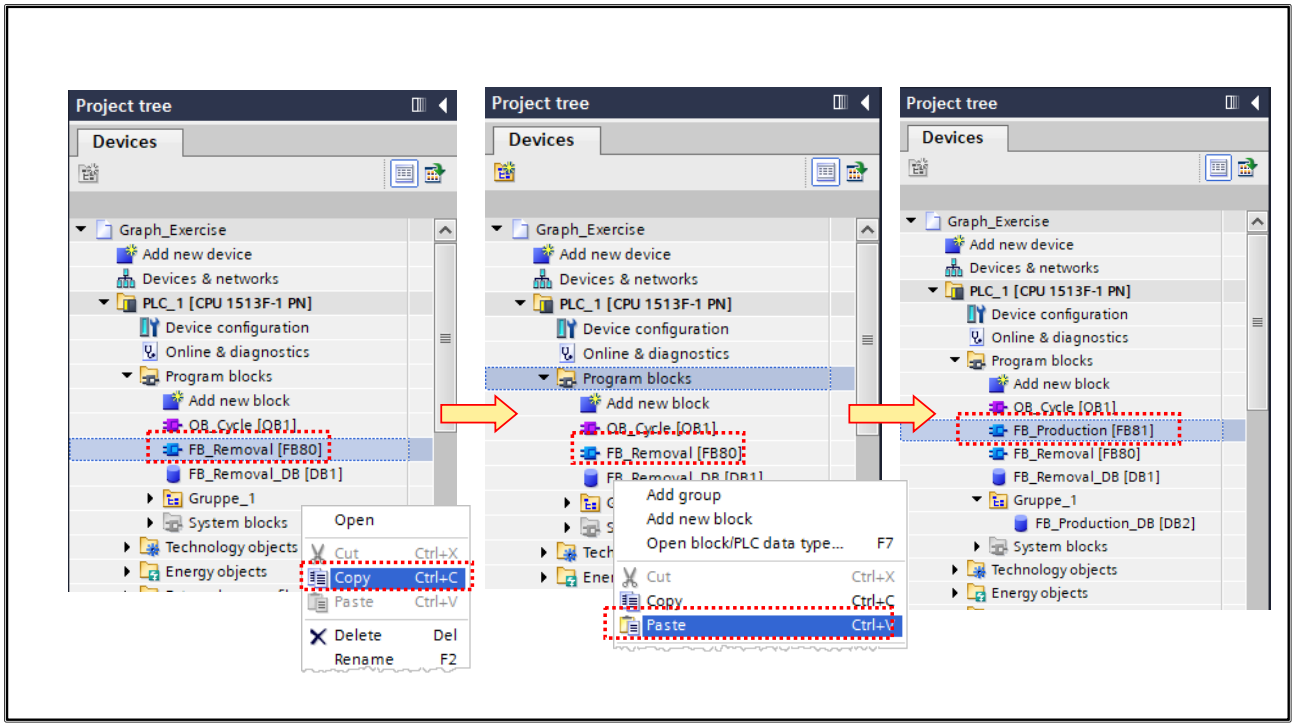
Expansion of the sequence from Chapter 3 in the following functions:

- The sequence is to be expanded to include the above-shown steps.
- "Drilling" may only occur when pushbutton "S_Bay3" (%I3.3) is pressed and proximity switch "B_Ini3" (%I3.7) is occupied. The active drilling operation is indicated to the user by the lit LED "P_Bay3" (%Q3.3). The drilling operation must have an **uninterrupted** duration of 3 s! An advancement is only possible when an interlock is not present!
- "Milling", like "Drilling", is performed by pressing the associated pushbutton "S_Bay2" (%I3.2) when proximity switch "B_Ini2" (%I3.6) is occupied, and is signaled by the LED "P_Bay2" (%Q3.2). In contrast to "Drilling", however, "Milling" has no fixed prescribed uninterrupted processing time and an advancement to the next step can occur after a step duration 3 s if no interlock is present!
- The conveyor run is to be time-monitored when the conveyor is moving toward the left (6 s as value for monitoring time). If a transport operation exceeds this duration, an alarm is triggered.
  Another optional time monitoring is available for the step in which a new blank is placed on the conveyor.
- Ensure that the conveyor motor is switched off immediately when supervision errors occur.
- Acknowledge active supervision errors using the function block (parameter wiring with %I0.7 "S_Ack_Fault"). If necessary, correct the test settings for the step sequence. Place the input in the "Simulator" table.
- The "Acknowledge" step is no longer needed. Its acknowledgment function is integrated in the "Removal" step.

Procedure

Program the other steps as well as the necessary additions (Interlock and Supervision) in the single step view for the relevant steps and commission the program again.

4.12. Exercise 1: Create a new GRAPH FB

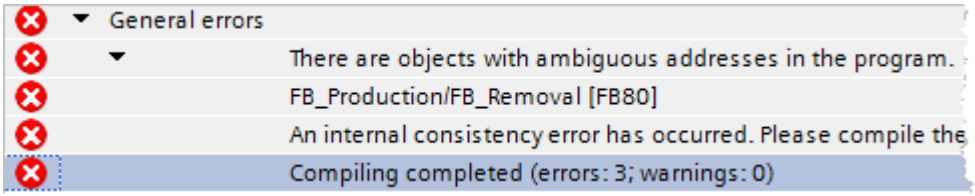


Task

Instead of creating a new FB or expanding your old FB, copy this and make the change to the copied FB.

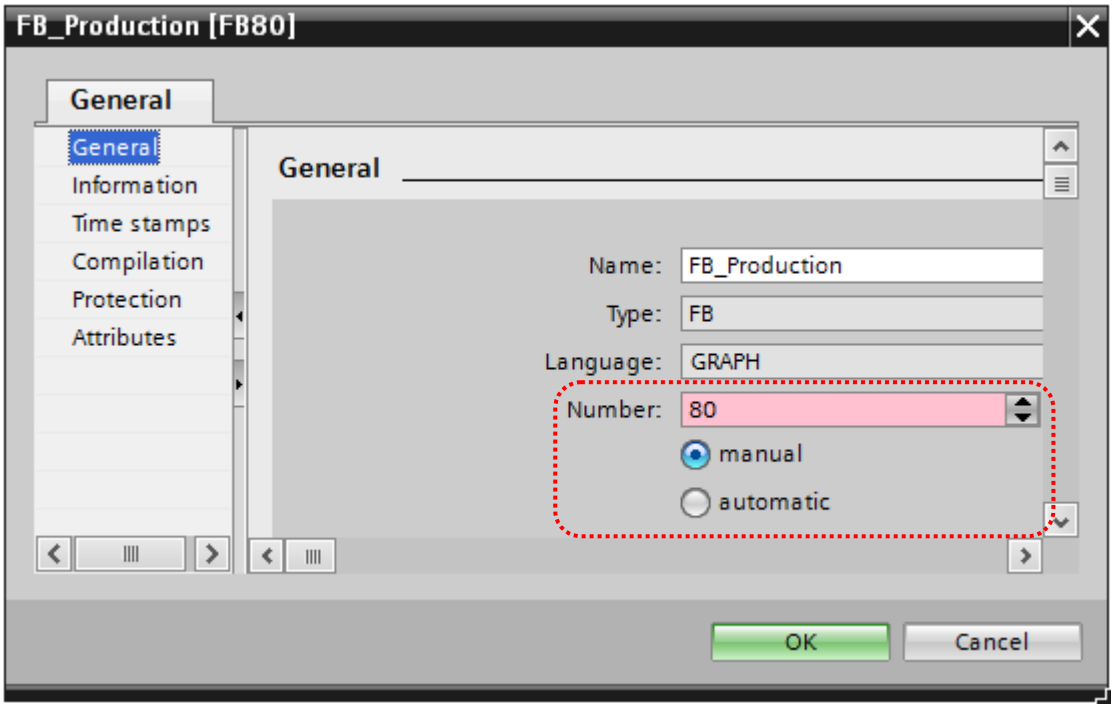
Procedure

Copy your FB80 created in the previous exercise and rename it "FB_Production". A compilation of the program blocks should now generate the following compiler errors:

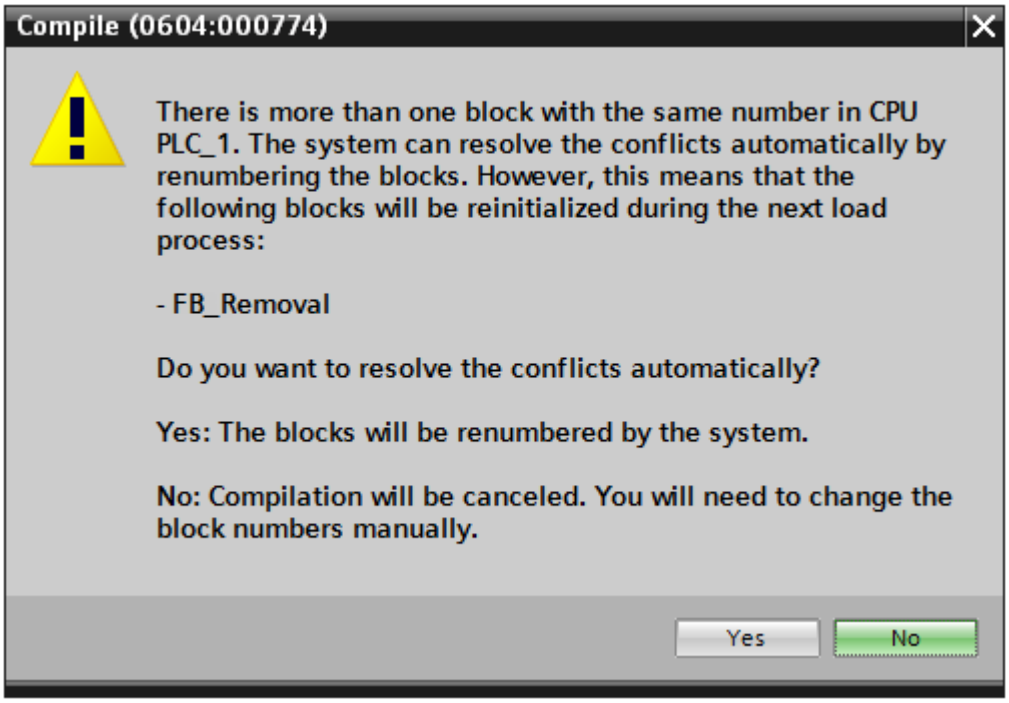

General errors
There are objects with ambiguous addresses in the program.
FB_Production/FB_Removal [FB80]
An internal consistency error has occurred. Please compile the
Compiling completed (errors: 3; warnings: 0)

For this reason, change the numbering assignment from "manual" to "automatic" in the general block properties of the copied block (right click on the FB -> Properties…).

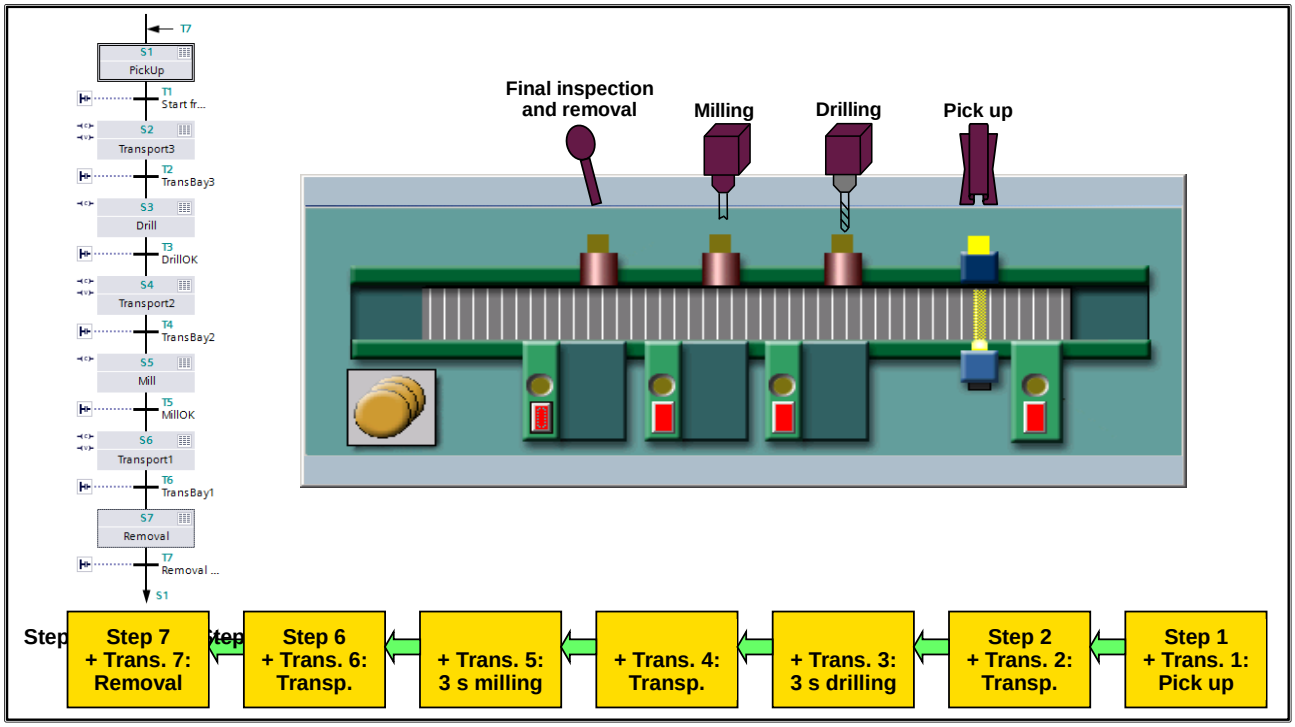
Another compilation of the program blocks of the PLC should now automatically adapt the number of the copied FB to the next lowest available range.



If you try to download the block to the PLC without compiling first, the TIA Portal will display the following dialog:


Compile (0604:000774)
There is more than one block with the same number in CPU PLC_1. The system can resolve the conflicts automatically by renumbering the blocks. However, this means that the following blocks will be reinitialized during the next load process:
- FB_Removal
Do you want to resolve the conflicts automatically?
Yes: The blocks will be renumbered by the system.
No: Compilation will be canceled. You will need to change the block numbers manually.
Yes
No

4.13. Exercise 2: Expand the conveyor functionality



Task

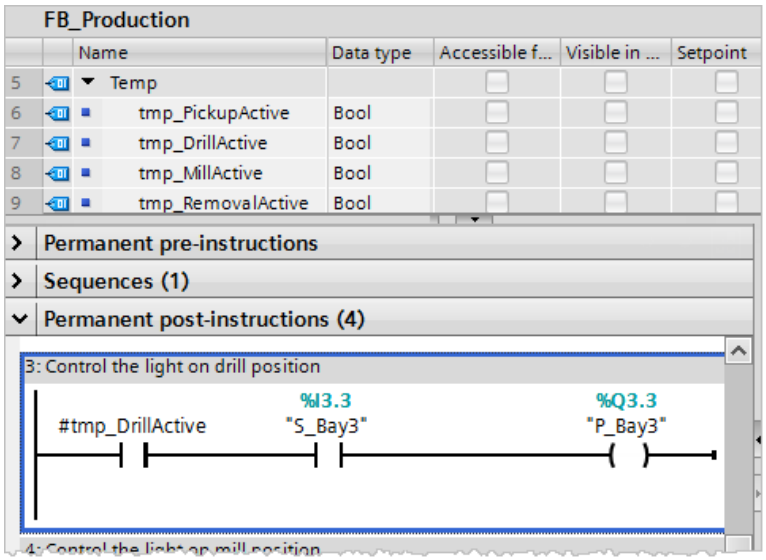
The new steps are to satisfy the functionality called for in the task. The existing "Pick up" and "Removal" steps remain unchanged for the most part.

Procedure

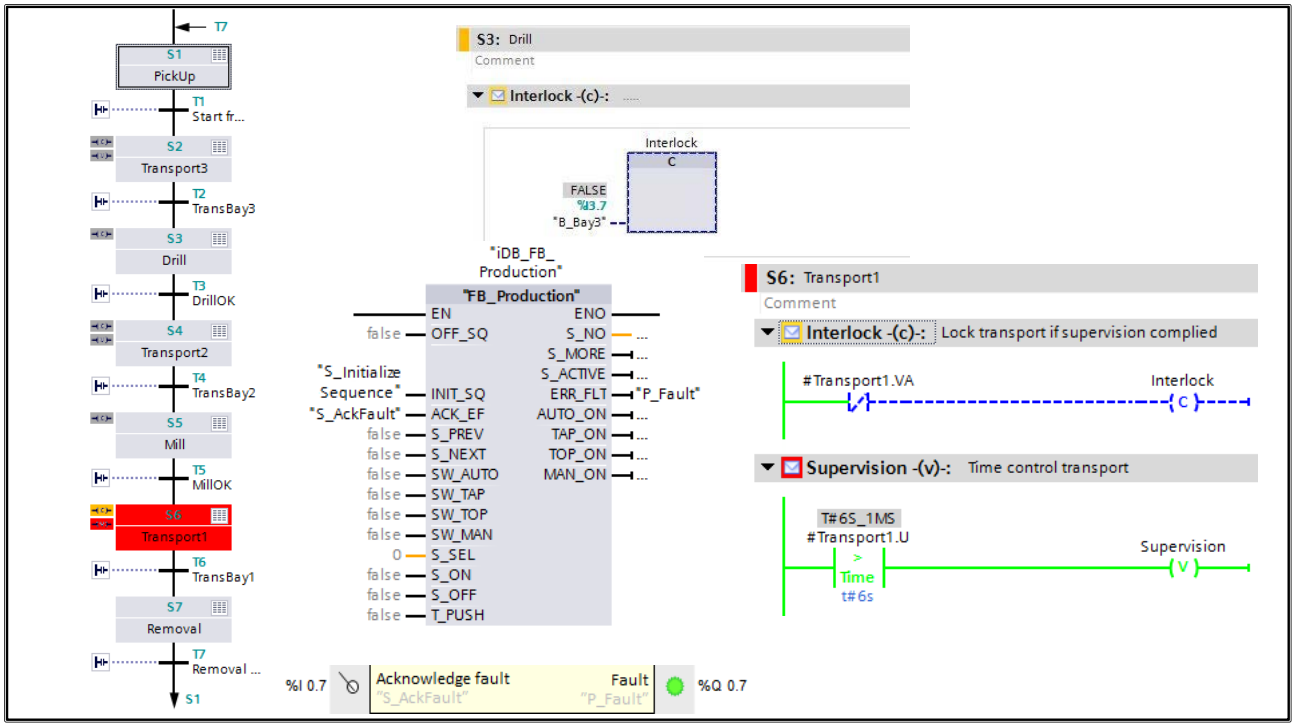
Program the necessary additions (Interlock and Supervision) in the single step view for the relevant steps and commission the program again ("Automatic" mode).

Note

It is recommended that all illumination functions be programmed in the permanent post-instructions. The conditions can be logically combined there with, for example, static tags, that are set in the respective steps.



4.14. Exercise 3: Interlock and Supervision



Task

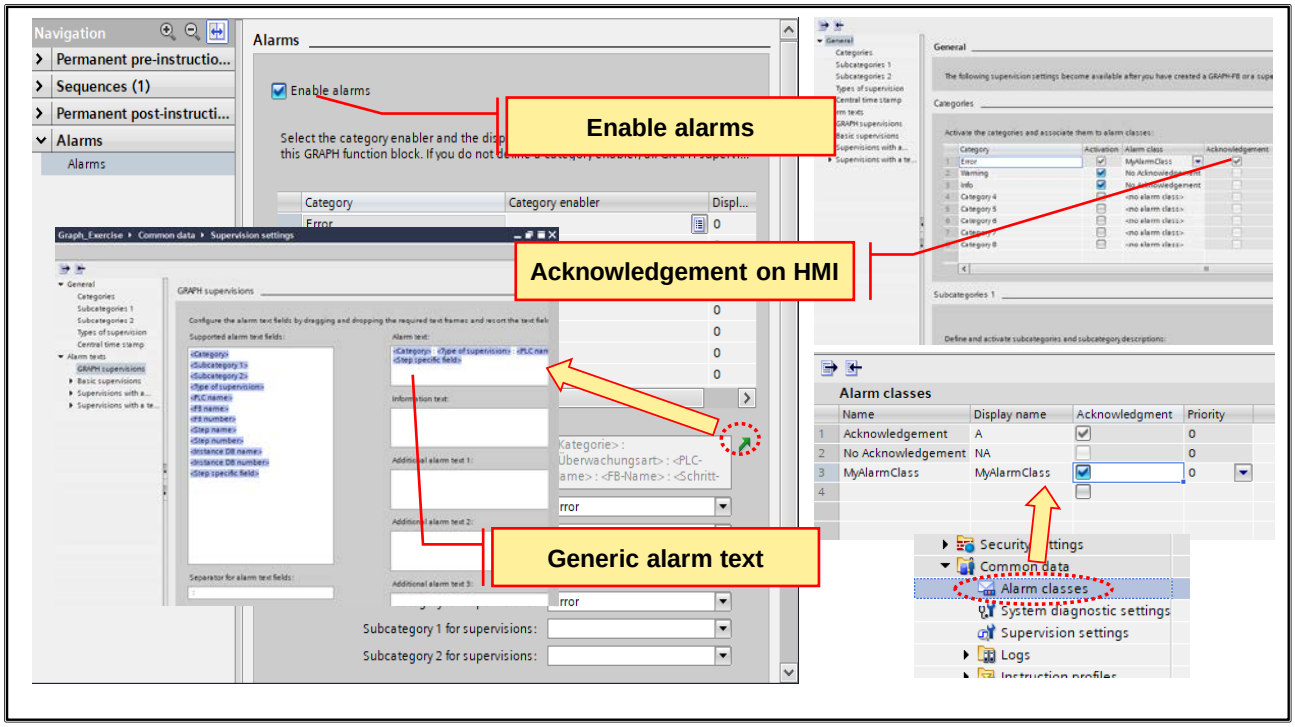
Program the interlocks as well as the supervisions of the transport steps called for in the task.

In addition, add the input "S_AckFault" (%I0.7) for the fault acknowledgment and the output "P_Fault" (%Q0.7) for the fault display for the call of the GRAPH FB in OB1.
You create these in the PLC tag "Simulator".

Note

The interlock conditions for advancement off the "Drilling" and "Milling" steps are different.
An interlock is also used to switch off the conveyor motor.
The appropriate step time must be selected for the monitoring the conveyor running time! The step activation time "T" and the uninterrupted step activation time "U" have a different response to faults!

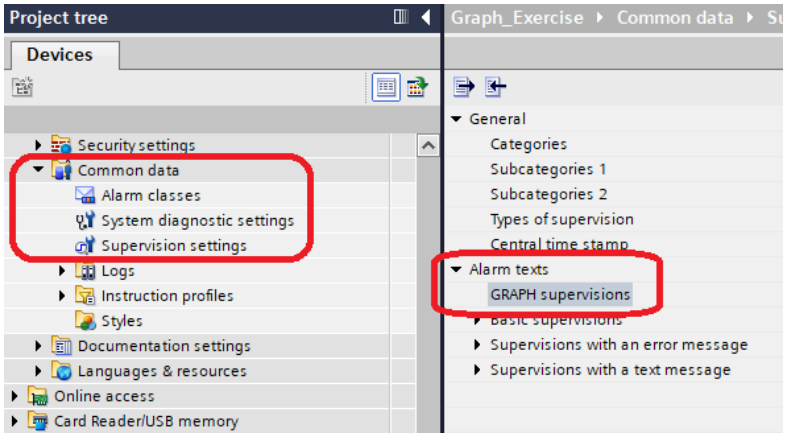
4.15. Configuring alarms



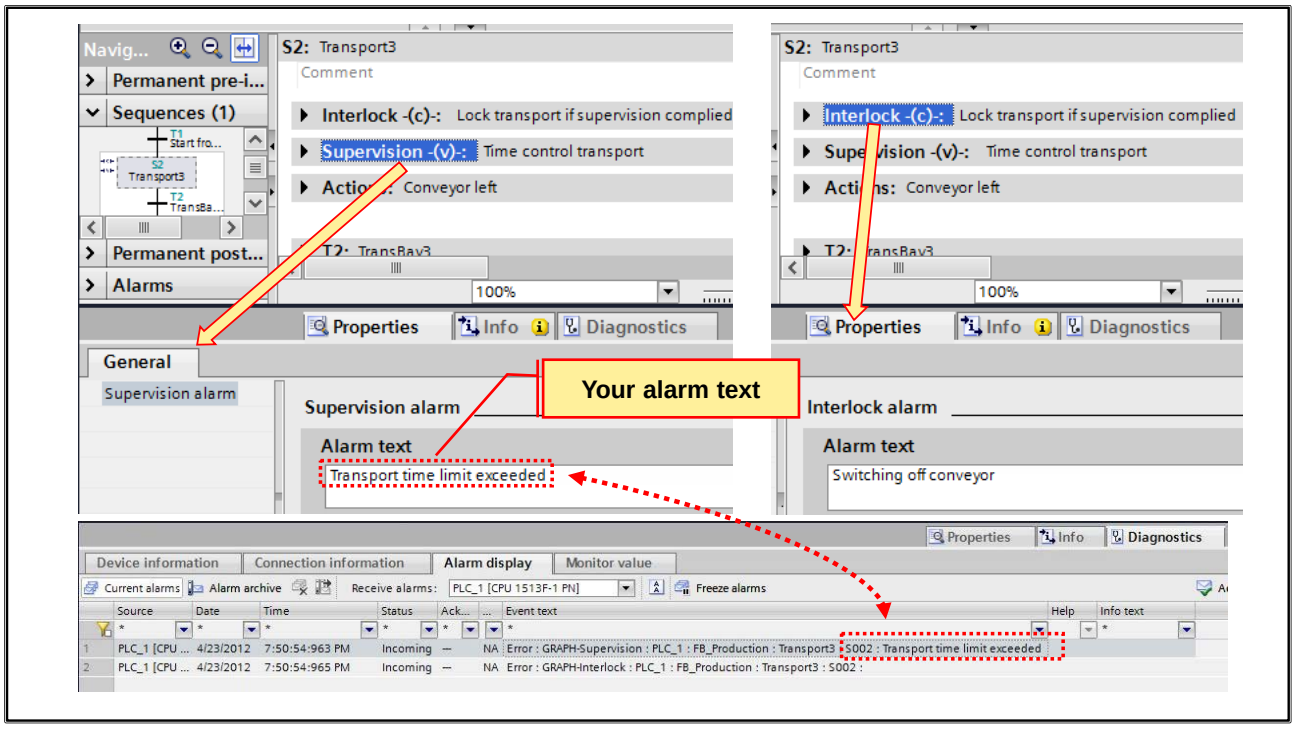
Alarms

Defines the alarm that is output for a step when an interlock condition is false / supervision condition is true. The same alarm structure is used for all steps. The default text with white background can be changed. The information fields with gray background are filled out automatically under error conditions. Associated values cannot yet be supplied with these alarms.

The name of the alarm class is language-neutral and cannot be changed. The display name can be edited.
Supervision settings and alarm classes are managed at a central place in the project tree, under “Common data”.



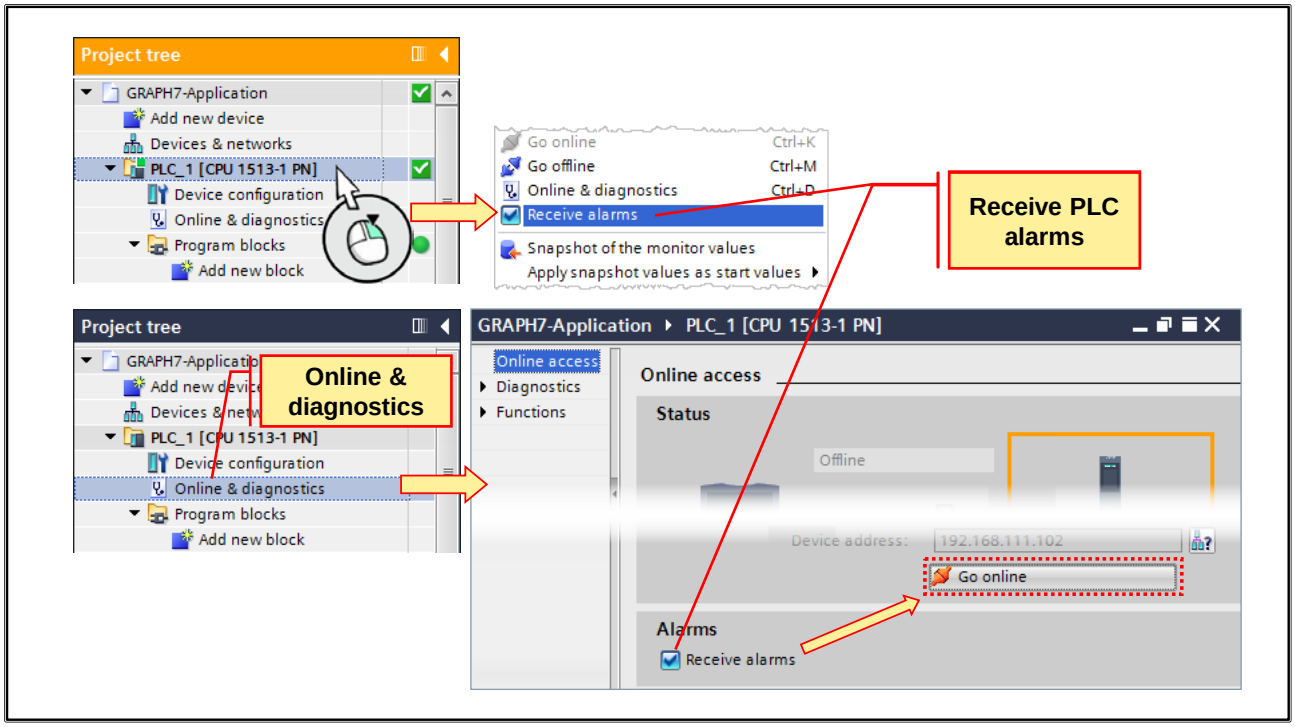
4.15.1. Adding step-specific alarm texts



Alarms

A step-specific text can be created in each step using the "General properties" of the interlock and supervision. This text is dependent on the alarm for interlocks and supervisions if this property is activated.

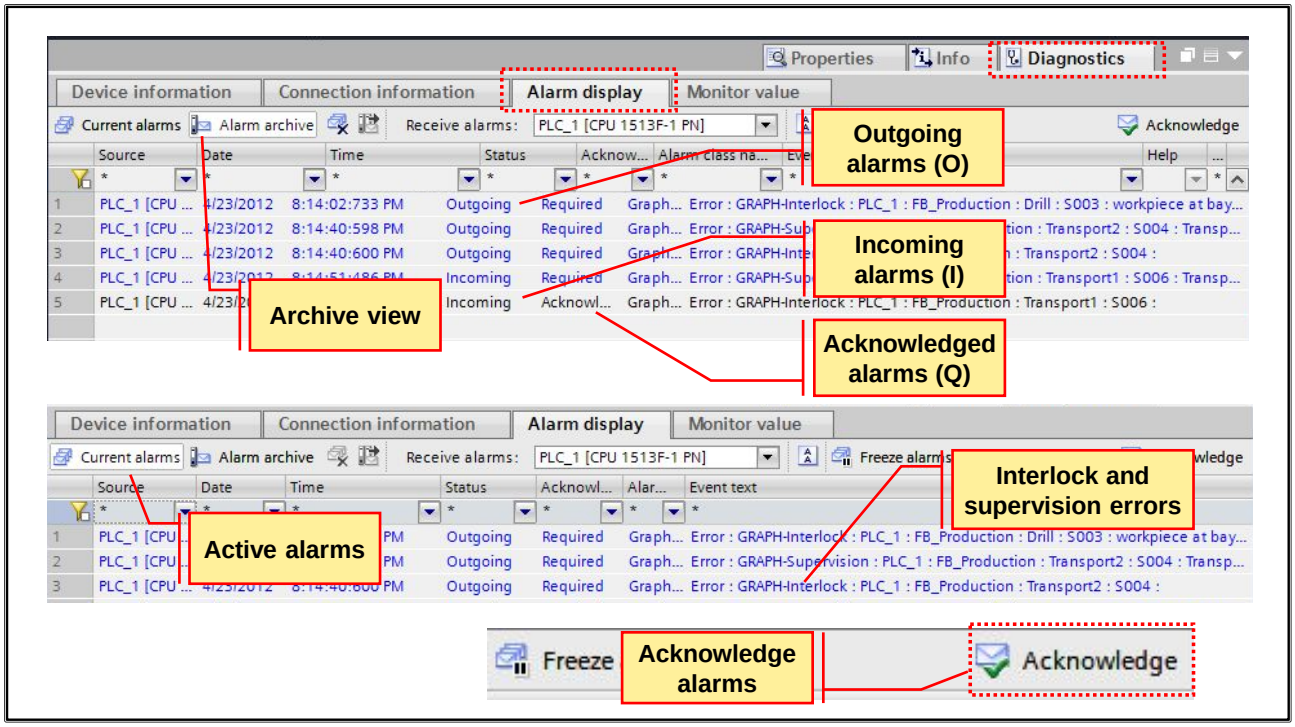
4.16. PLC alarms for displaying sequence errors



PLC alarms

The PLC alarms can be activated in the online display using the shortcut menu of the CPU. To do so, select the "Receive alarms" check box. Alternatively, the alarms can also be activated in the Online & diagnostics view.

4.17. Display and acknowledgement

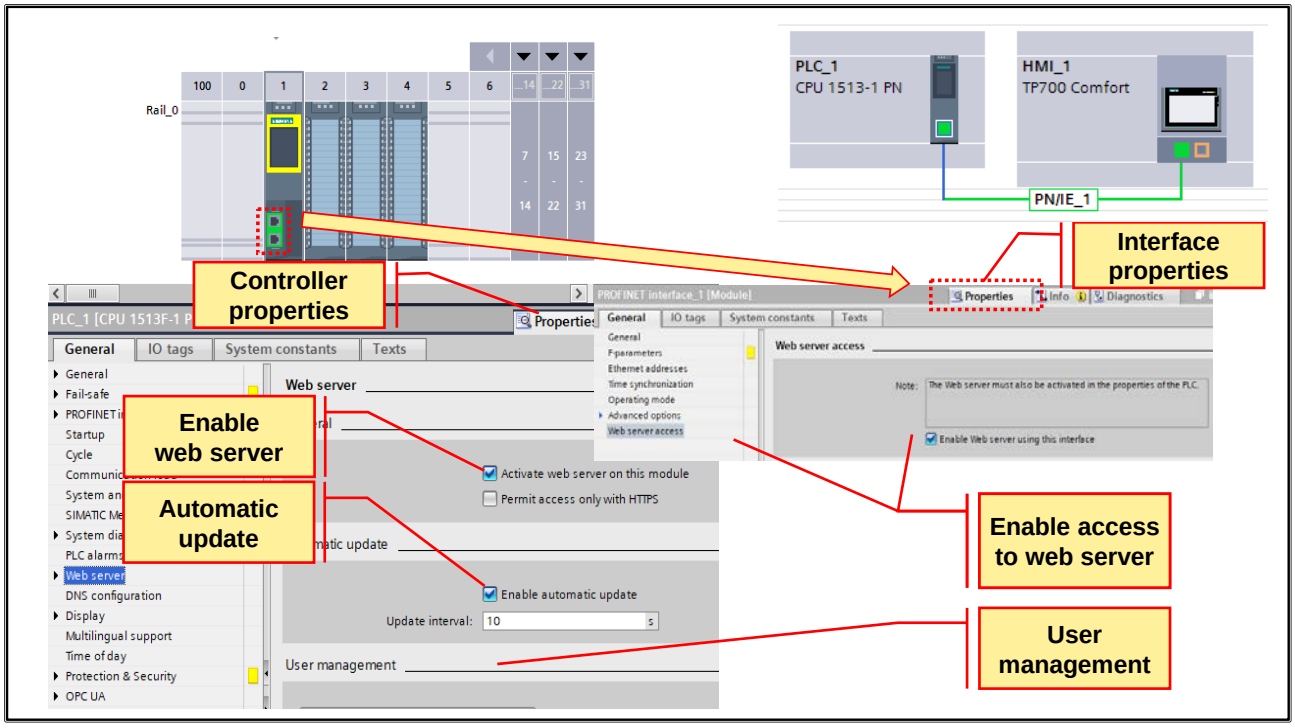


Alarm display

The interlock and supervision error messages with time stamp generated by GRAPH are displayed in the alarm display. The alarm display menu allows acknowledgment of alarms using the "Acknowledge alarms" icon.

The acknowledgment does not cause the interlock or supervision error to be canceled. This must be done, same as before, using the corresponding FB input parameter ACK_EF. Acknowledgment of an "Active alarm" moves this alarm to the "Archive view", however.

4.18. Activating the web server on the CPU



Activating the web server

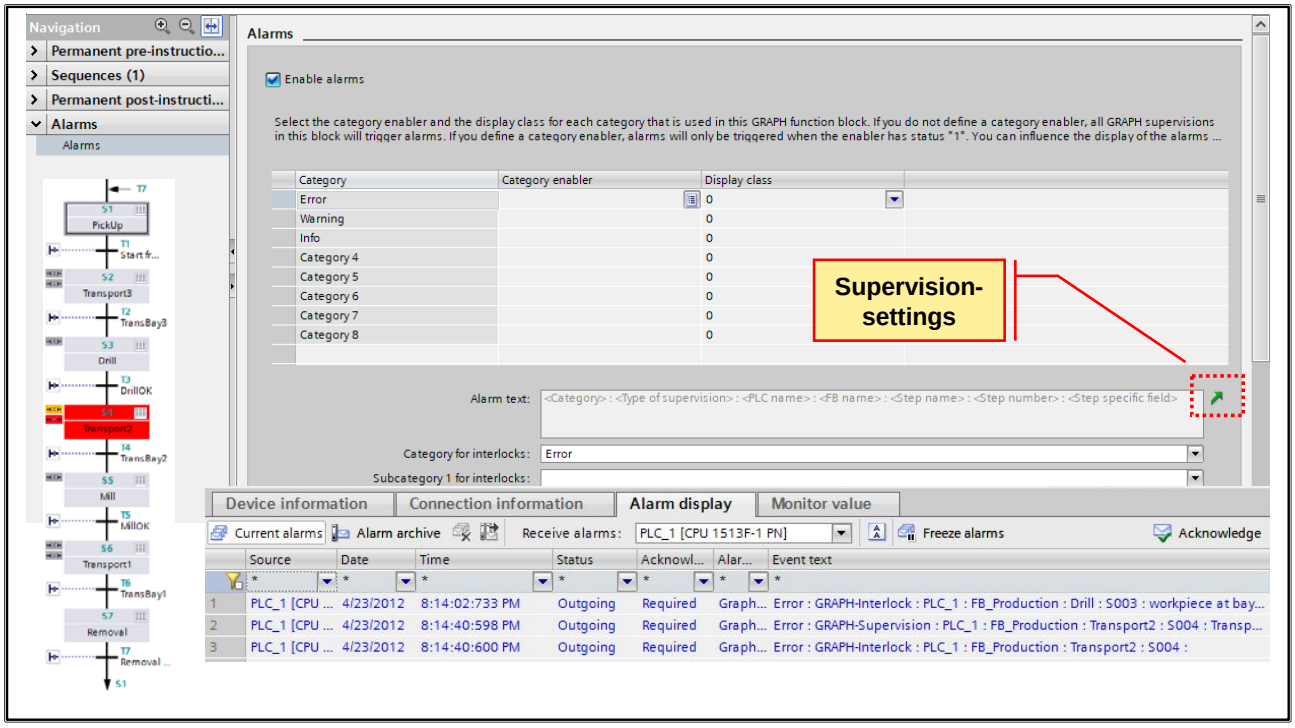
The web server is activated on the PROFINET CPU with a few mouse clicks:

- Select CPU in the device or network view
- Select web server in the CPU properties
- Activate the web server and set the update interval to 10 s
- Activate project languages and language in the web server
- Compile and download configuration

Note

In order to access the web server, the access via the interface you want to use to reach the Web server of the PLC must also be activated!

4.19. Exercise 4: Display and acknowledge PLC alarms


Navigation
Permanent pre-instructio...
Sequences (1)
Permanent post-instructi...
Alarms
Alarms
Alarms
Enable alarms
Select the category enabler and the display class for each category that is used in this GRAPH function block. If you do not define a category enabler, all GRAPH supervisions in this block will trigger alarms. If you define a category enabler, alarms will only be triggered when the enabler has status "1". You can influence the display of the alarms ...
Category
Category enabler
Display class
Error
Warning
Info
Category 4
Category 5
Category 6
Category 7
Category 8
Supervision-settings
Alarm text:
<Category> : <Type of supervision> : <PLC name> : <FB name> : <Step name> : <Step number> : <Step specific field>
Category for interlocks:
Error
Subcategory 1 for interlocks:
Device information
Connection information
Alarm display
Monitor value
Current alarms
Alarm archive
Receive alarms:
PLC_1 [CPU 1513F-1 PN]
Freeze alarms
Acknowledge
Source
Date
Time
Status
Acknowl...
Alar...
Event text
PLC_1 [CPU ... 4/23/2012 8:14:02:733 PM Outgoing Required Graph... Error : GRAPH-Interlock : PLC_1 : FB_Production : Drill : S003 : workpiece at bay...
PLC_1 [CPU ... 4/23/2012 8:14:40:598 PM Outgoing Required Graph... Error : GRAPH-Supervision : PLC_1 : FB_Production : Transport2 : S004 : Transp...
PLC_1 [CPU ... 4/23/2012 8:14:40:600 PM Outgoing Required Graph... Error : GRAPH-Interlock : PLC_1 : FB_Production : Transport2 : S004 :

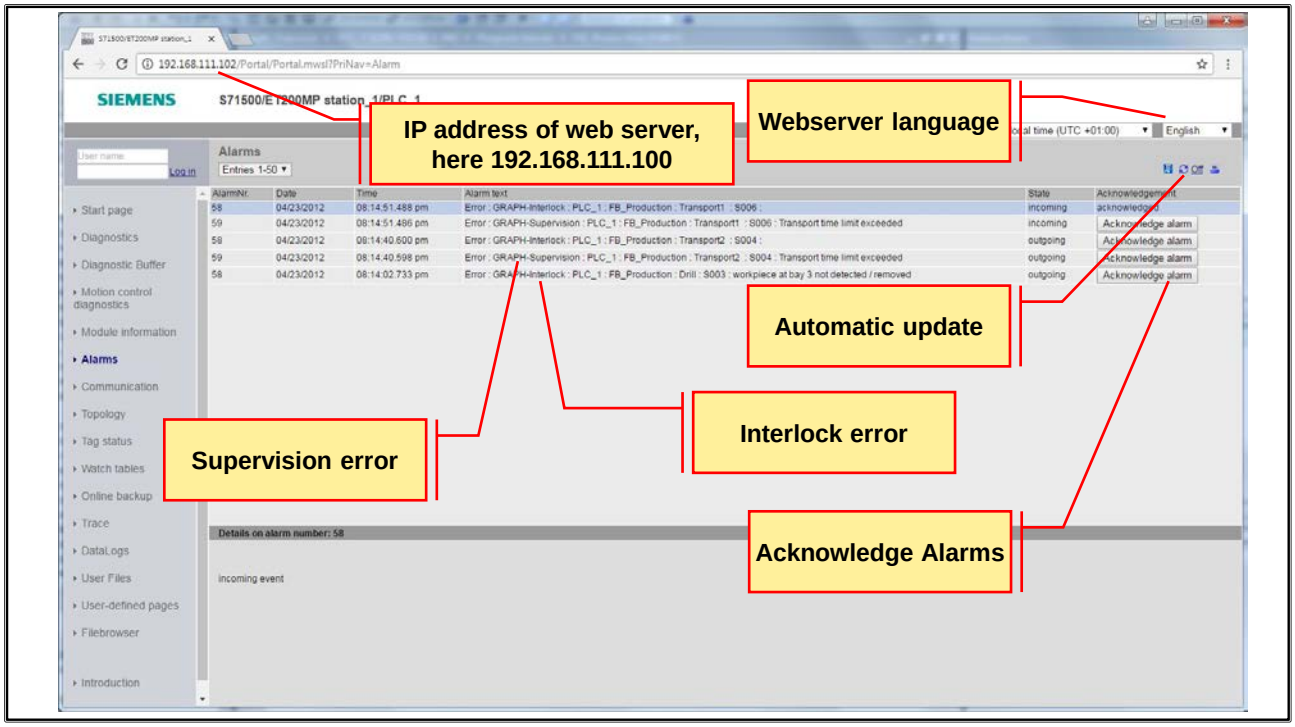
Task

When interlock and supervision errors occur, the alarms for interlock and supervision errors stored in the system are to be displayed as PLC alarms.

Procedure

1. Configure the alarm texts for interlock and supervision errors according to the preceding pages. Create your own alarm texts for the various interlock and supervision errors in the steps.
2. Enable the PLC alarms on your controller.
3. Open the alarm display and trigger interlock and supervision errors in your sequence.
4. Monitor the incoming alarms and acknowledge them.

4.20. Exercise 5: Display of sequence errors in the web server



Task

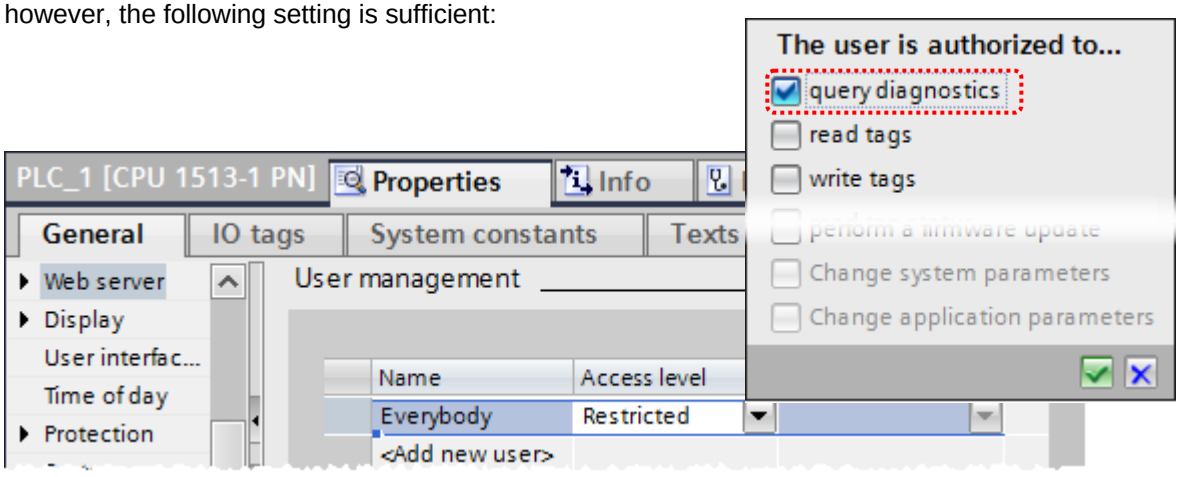
The error messages generated by interlock and supervision errors that were visualized previously using PLC alarms are to be displayed on the integrated web server of the CPU. In this way, any type of step sequence error can be monitored remotely in principle.

Procedure

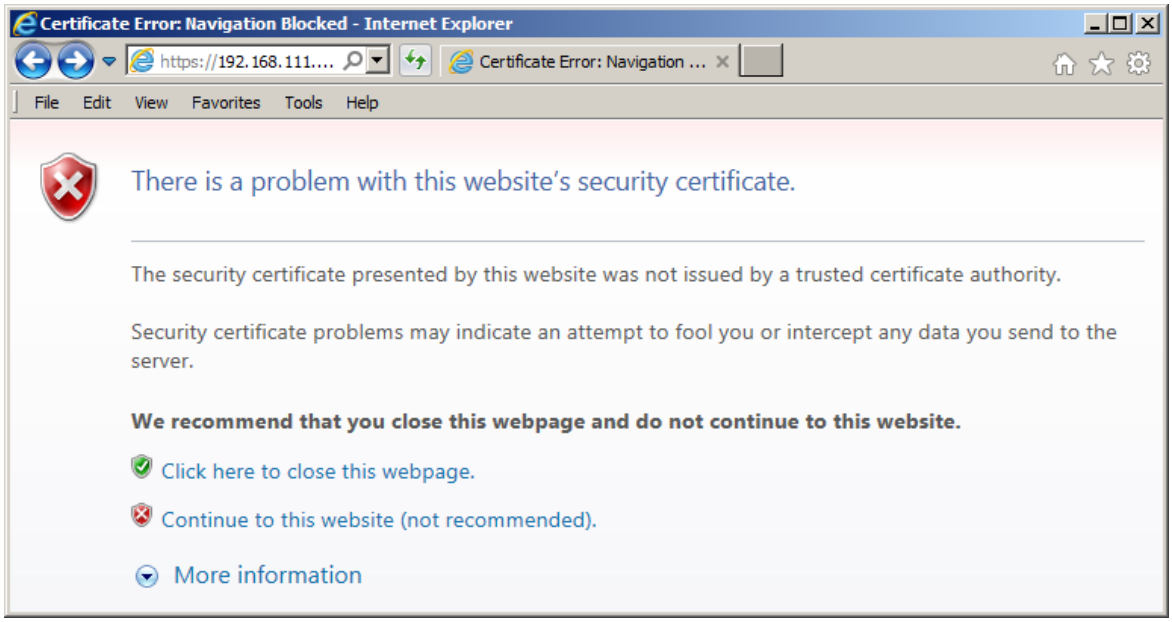
1. Activate the web server of your controller as shown on the theory page.
2. Call up the internal web server, e.g. via https://192.168.111.102 as the currently valid IP address.
3. Trigger interlock and supervision errors in your step sequence in the familiar way and monitor them in the web server on the "Alarms" tab.

Notes

In order to see the menu items shown in the figure in the web server, you must also set the corresponding access rights in the TIA Portal. The functions you have in the case of an administrative access in the Web server are shown at the top of the figure. For this exercise, however, the following setting is sufficient:

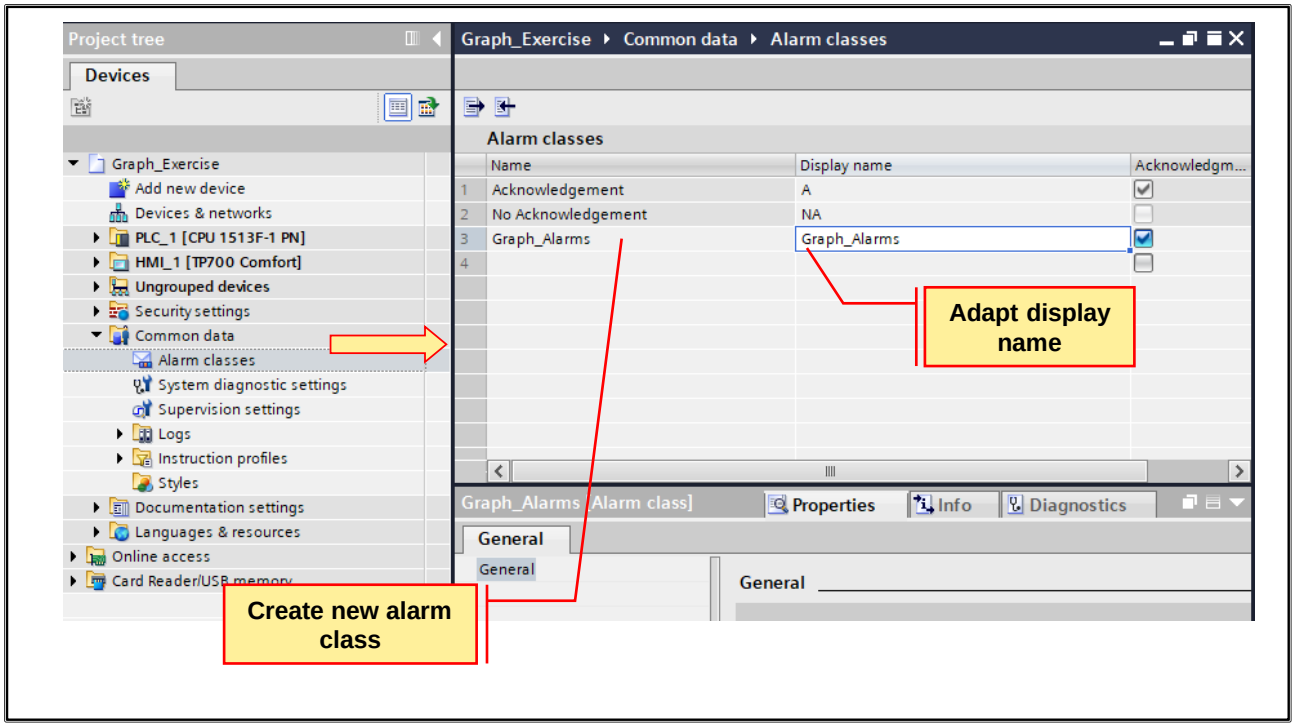


When the web server is accessed via https, you normally see the following page in your browser:


Certificate Error: Navigation Blocked - Internet Explorer
https://192.168.111....
Certificate Error: Navigation ...
File Edit View Favorites Tools Help
There is a problem with this website's security certificate.
The security certificate presented by this website was not issued by a trusted certificate authority.
Security certificate problems may indicate an attempt to fool you or intercept any data you send to the server.
We recommend that you close this webpage and do not continue to this website.
Click here to close this webpage.
Continue to this website (not recommended).
More information

te (not recommended)" Click to continue to this website of the SIMATIC S7-1500.

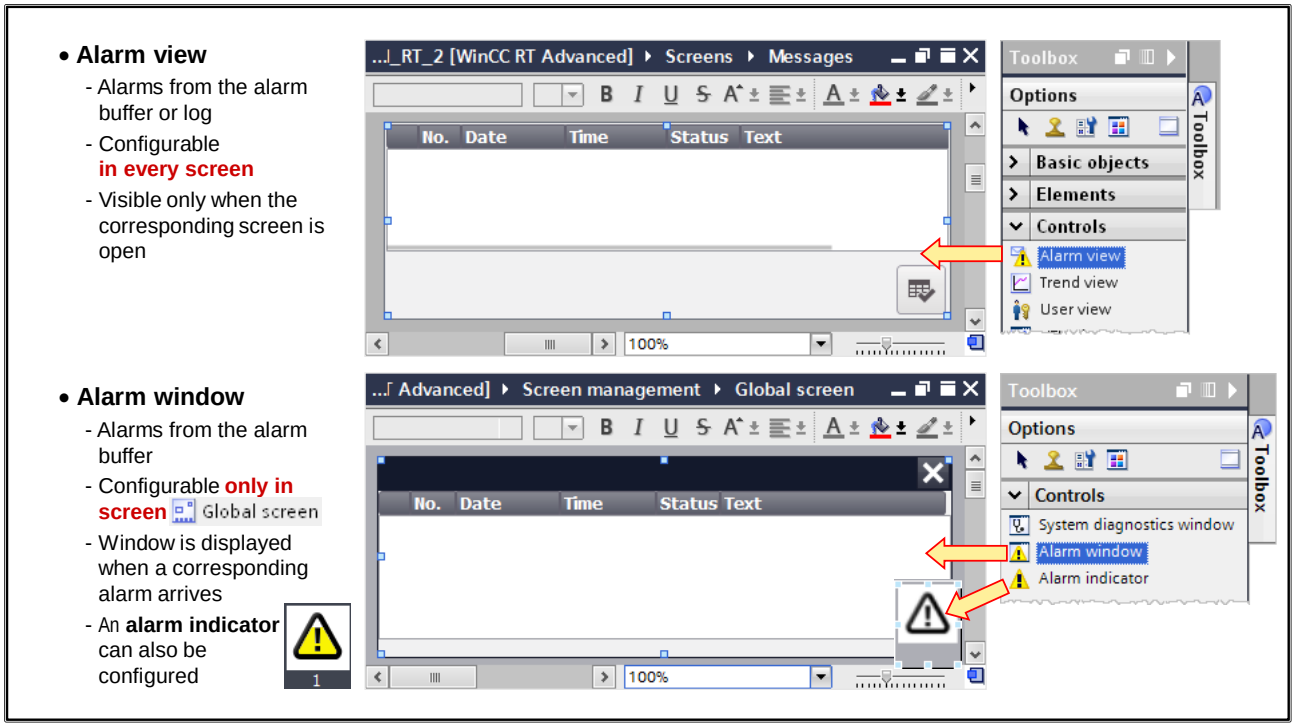
4.21. Creating your own alarm class for the project



General

The user has the option of creating his/her own alarm classes. These can then be used in all elements of the project.

4.22. Displaying alarms on the HMI device



General

Multiple alarm displays for different alarm classes and in different screens and windows can be configured. Two variants are distinguished for displaying:

Alarm view

- Shows currently active alarms or alarms from the alarm buffer or alarm log.
- Selection of alarms according to alarm class.
- Can be configured in any screen.
- Only visible when the corresponding screen is open.
- Choice of display type: Table | Line.
- An alarm view gives the user the following options at runtime.

- **Tooltip**: Shows the tooltip for the selected alarm.

- **Loop in Alarm:** If a screen change is configured for the selected alarm, the display switches to a screen containing information about the alarm.

- **Acknowledge:** Acknowledges the selected alarm.

Alarm window

- Shows currently active alarms or alarms from the alarm buffer or alarm log.
- Selection of alarms according to alarm class.
- Can only be configured in the "Global screen" screen.
- Window can be automatically displayed when a new entry is made in the window (e.g. new incoming message).

Alarm indicator

As an alternative to the automatic display of an alarm window, this small graphic symbol can be displayed symbolizing the arrival of displayable alarms (group alarms) of a defined alarm class. This alarm indicator can have two states:

Flashing display:
At least one unacknowledged alarm is pending.

Static display:
All alarms with mandatory acknowledgment are acknowledged but at least one of them does not yet have outgoing status.

The number shown at the bottom of the symbol indicates the number of alarms that are still active.

4.23. Alarm components and properties

Basic components:

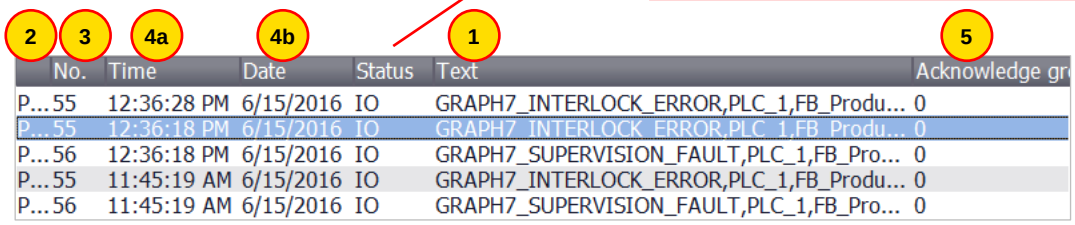
1 - Alarm text (with maximum of 8 process values, if necessary)
2 - Alarm class for alarm properties
(with optionally configurable display name for the alarm class)
3 - Alarm number
(unique within an alarm procedure)
4 - Time stamp (when the alarm was generated)
5 - Alarm group (for group acknowledgement)

Alarm statuses without mandatory acknowledgment

I	Incoming
IO	Outgoing

Alarm statuses with mandatory acknowledgment

I	Incoming
IO	Outgoing and not yet acknowledged
IOA	Outgoing and acknowledged afterwards
IA	Outgoing and acknowledged
IAO	Outgoing and acknowledged beforehand



Properties

- Mandatory acknowledgement On/Off (default setting: Off)
- Color design of alarm display
- Alarm status (I - Incoming; O
- Archiving On/Off

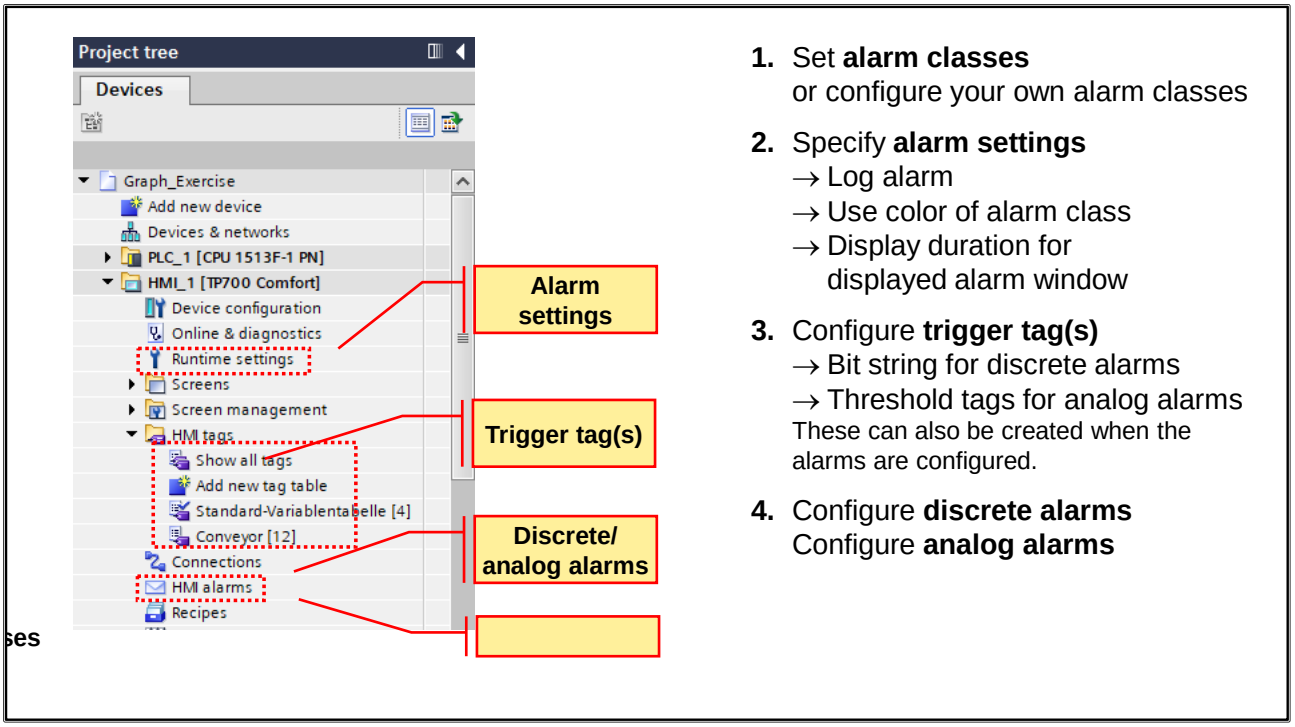
edged)

Tasks

The alarm system of an HMI device performs the following tasks:

- **Displaying**
  Configured alarms are displayed on the HMI device dependent on events or statuses that occur in the plant or process.
  ⇒ Functionality is supported by every HMI device
- **Reporting**
  The alarms are output to a printer.
  ⇒ Functionality is supported only by HMI devices that also support reports
- **Archiving**
  The alarms are stored for further processing and evaluation in a file (in Flash memory or on the hard disk).
  ⇒ Functionality is supported only by HMI devices that also support archives
- **Sending e-mails**
  With the system function "SendEmail", incoming alarms can send emails. That is a functionality of the Runtime option "Sm@rt Service" and will not be covered in this chapter.
  ⇒ Functionality is supported only by HMI devices that also support an Ethernet interface

4.24. Steps for alarm configuration



Alarm class

Every alarm must be assigned to an alarm class. Most importantly, the alarm classes specify the mandatory acknowledgment and the appearance of alarms in the display on the HMI device. They are also used to group alarms for various display means.

WinCC has predefined alarm classes as well as the option to configure your own alarm classes.

Predefined alarm classes "Errors" and "Warnings"

These are created for use as desired.

Errors:
For discrete and analog alarms that indicate critical or dangerous operating and process states. Alarms of this alarm class must always be acknowledged.

Warnings:
For discrete and analog alarms that indicate regular operating and process states as well as process operations. Alarms of this alarm class provide information to the operator and do not require acknowledgment.

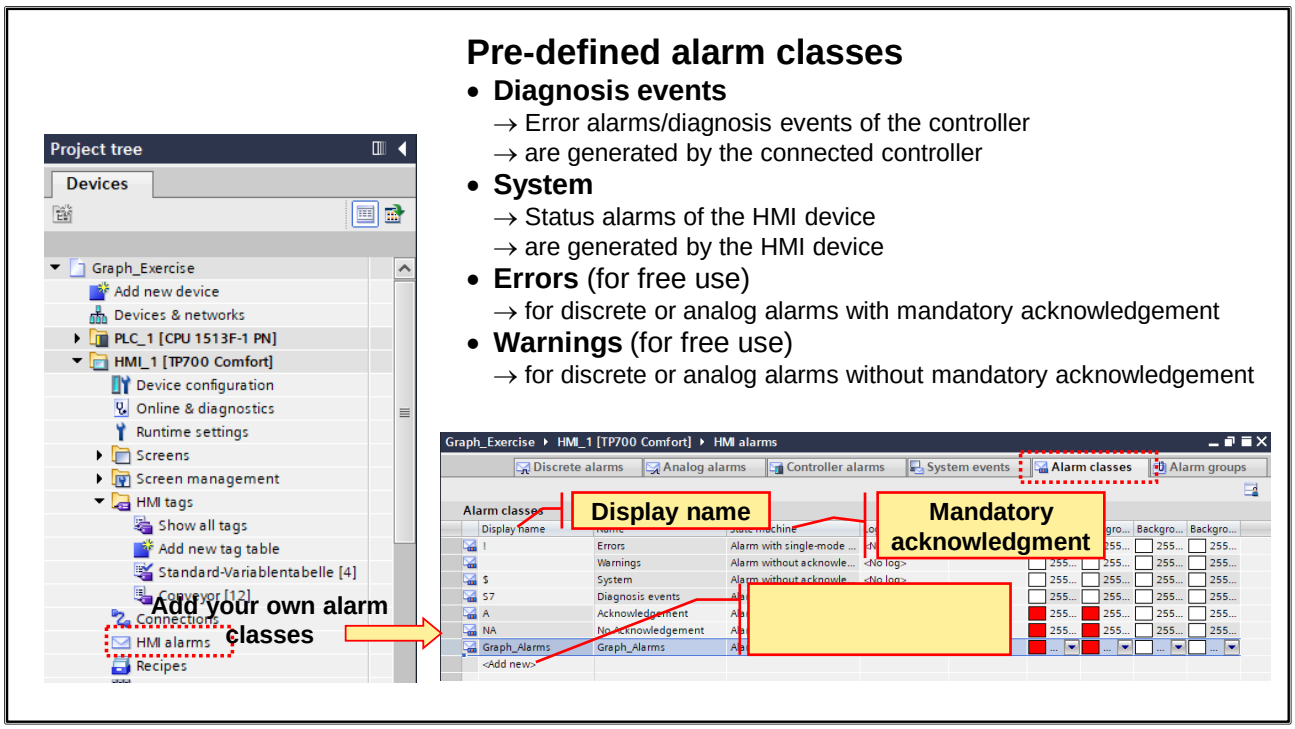
Display name

An individual name can be set here for each alarm class. This name can then be included in the alarm view when multiple alarm classes are shown in one alarm view. These display names, including for the predefined alarm classes, can be customized.

Acknowledging an alarm

Performed by the operator on the HMI device. The operator makes the acknowledgment using the acknowledgment key <ACK> (special key only on keyboard devices), function keys, softkeys or buttons. In addition, alarms can be acknowledged by system functions in function lists or scripts or by the controller program.

4.25. Alarm classes



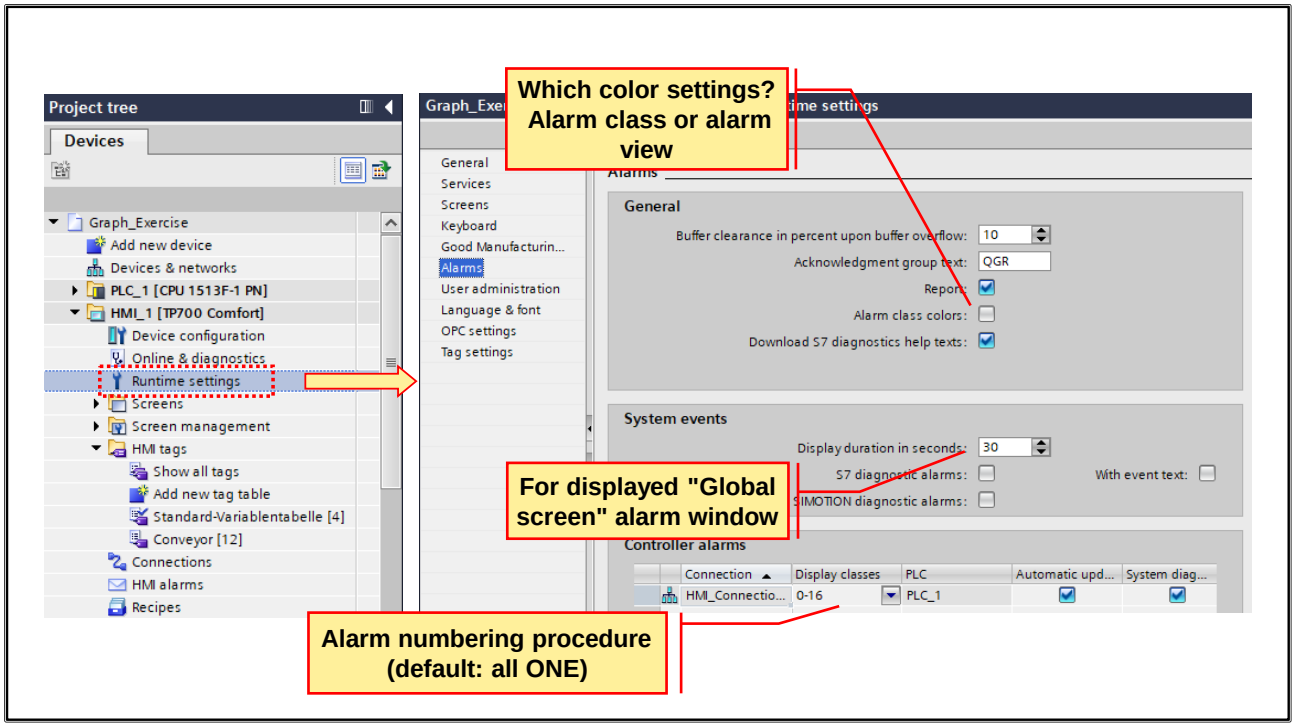
Alarm classes

Every alarm must be assigned to an alarm class. Most importantly, the alarm classes specify the mandatory acknowledgment and the appearance of alarms in the display on the HMI device. They are also used to group alarms for various display means.

WinCC has predefined alarm classes as well as the option to configure your own alarm classes.

Errors and warnings are predefined alarm classes.

4.26. Alarm settings



Settings

The alarm system of WinCC is also functional with the default settings, in principle. The changing of these settings serves simply to adapt the behavior of the alarm system to the respective circumstances of a plant.

System diagnostics

These are alarms that are sent by CPUs when this is enabled in the hardware properties, e.g. "CPU STOP due to programming error", "CPU STOP due to keyswitch operation".

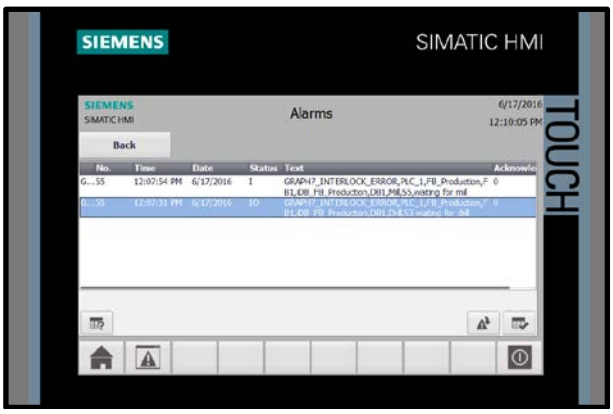
These alarms are not automatically displayed. You must always configure a display of the alarms yourself.

Controller alarms

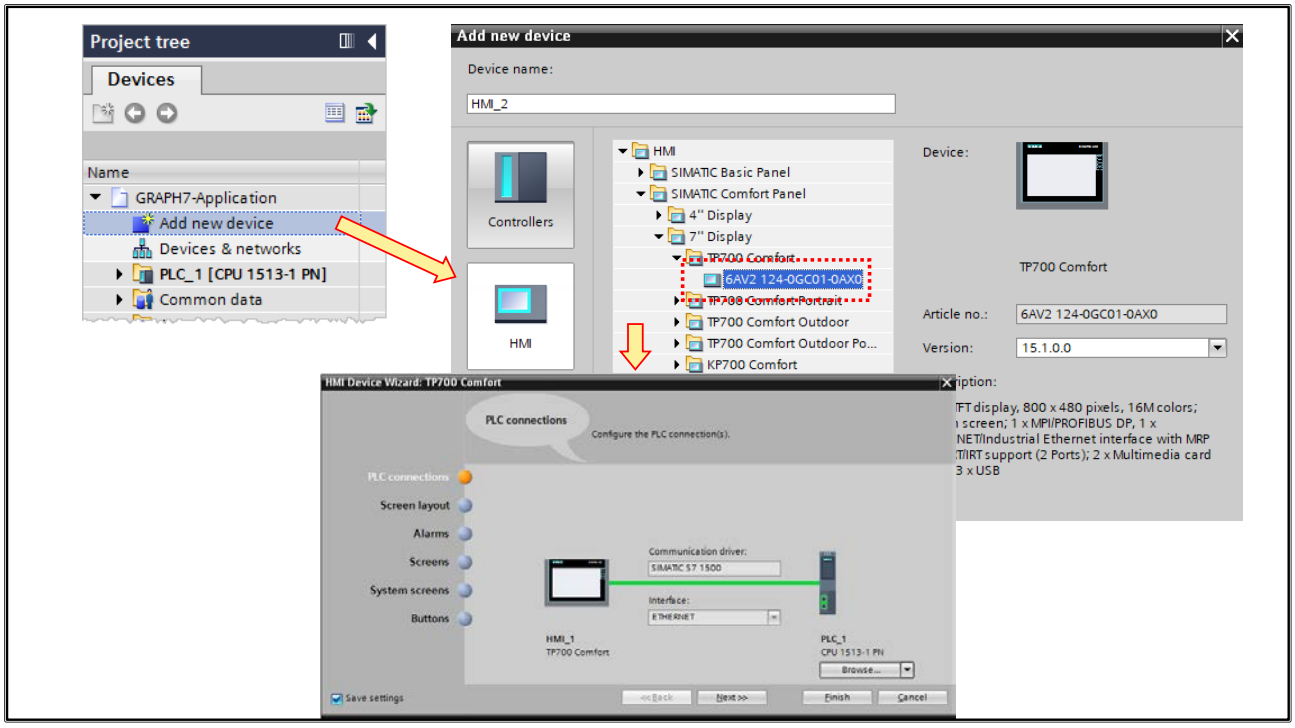
Settings for the alarm numbering procedure and system diagnostics (Program_Alarm functionality) are possible in this area.

Alarm view

The following figure shows the Runtime display of the alarm window on a TP700 Comfort containing the display of a step sequence error (Interlock error + Supervision error).



4.27. Exercise 6: Create touch panel

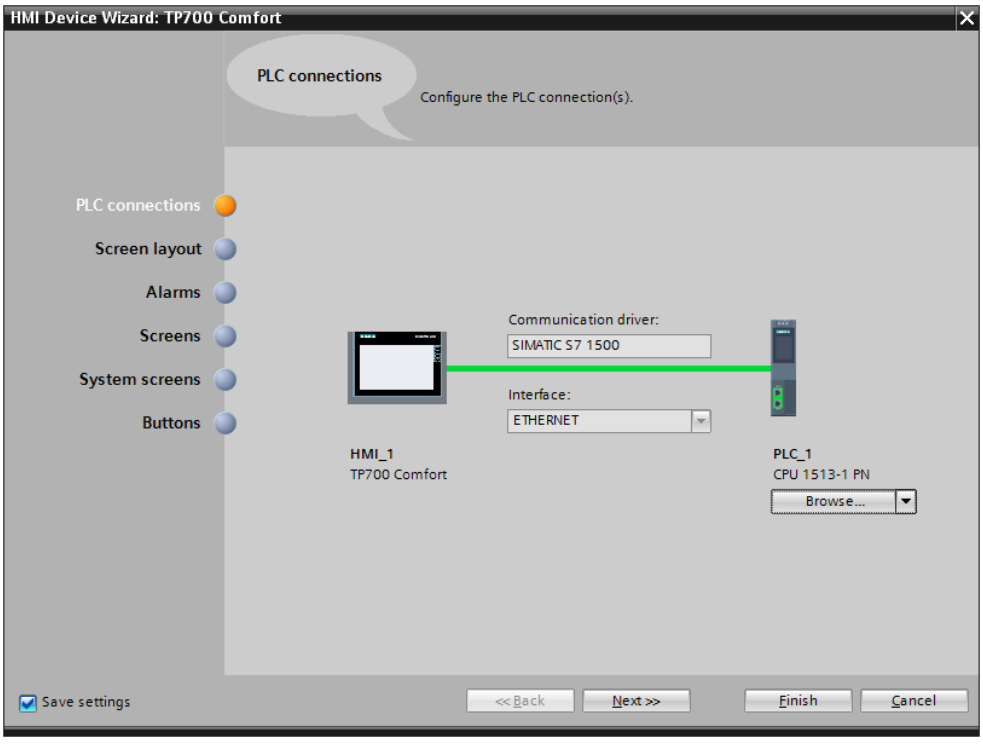


Task

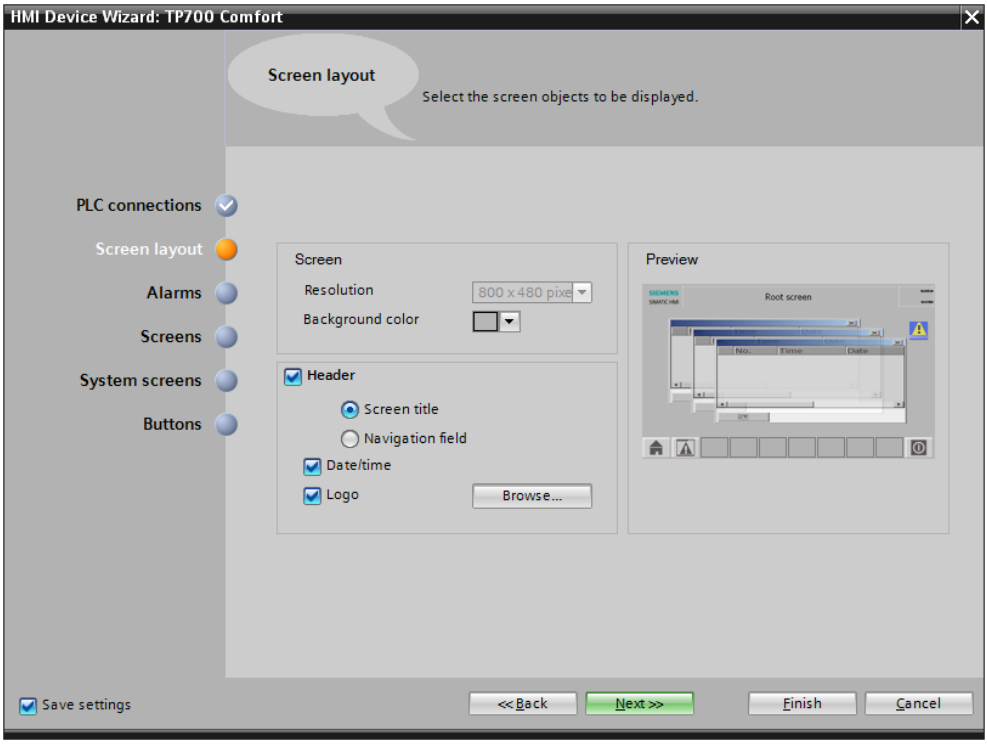
Create an HMI device "TP700 Comfort" using the wizard as shown in the following:
Be sure to disconnect the online connection to the PLC before performing this task!

Procedure

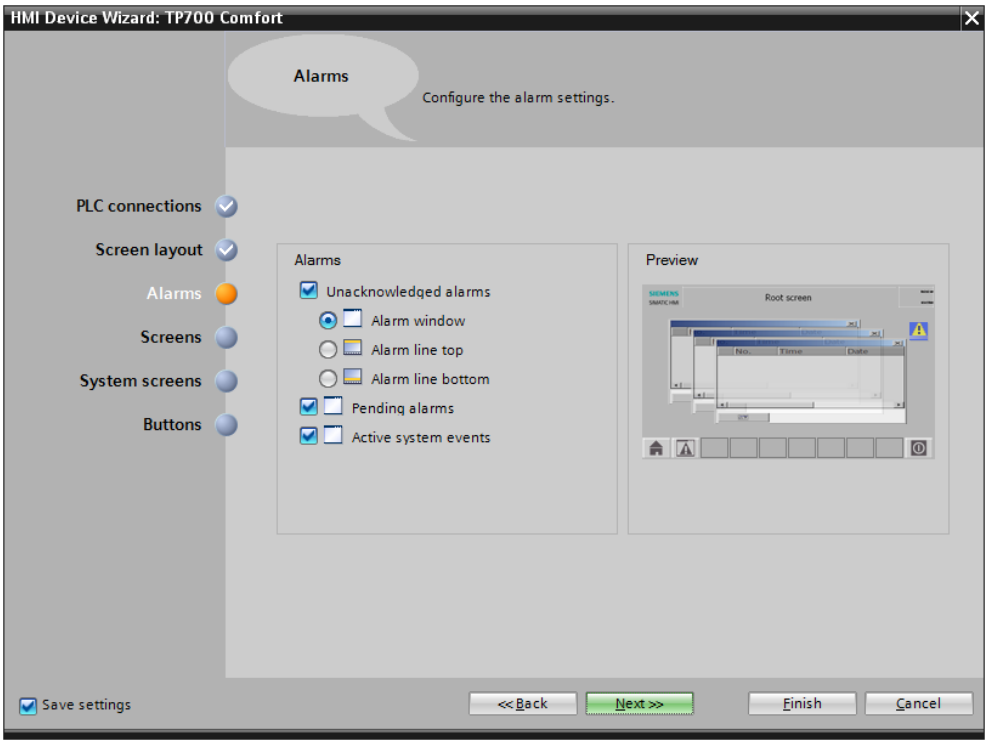
1.



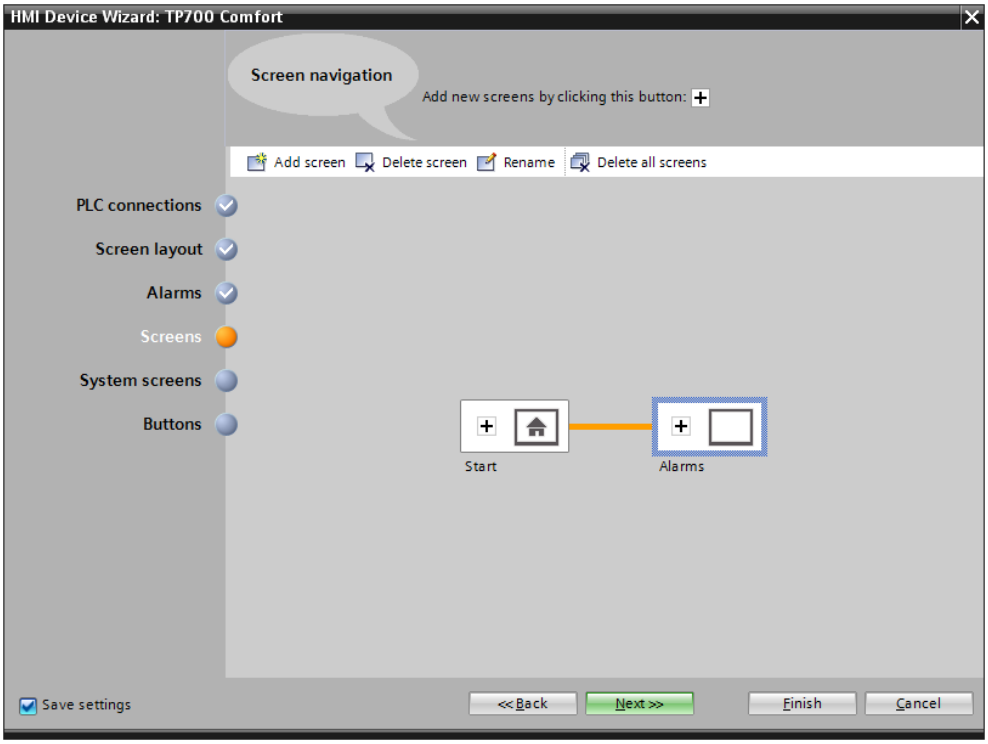
2.


HMI Device Wizard: TP700 Comfort
Screen layout
Select the screen objects to be displayed.
PLC connections
Screen layout
Alarms
Screens
System screens
Buttons
Screen
Resolution
800 x 480 pixe
Background color
Header
Screen title
Navigation field
Date/time
Logo
Browse...
Preview
Root screen
Save settings
<<Back
Next >>
Finish
Cancel

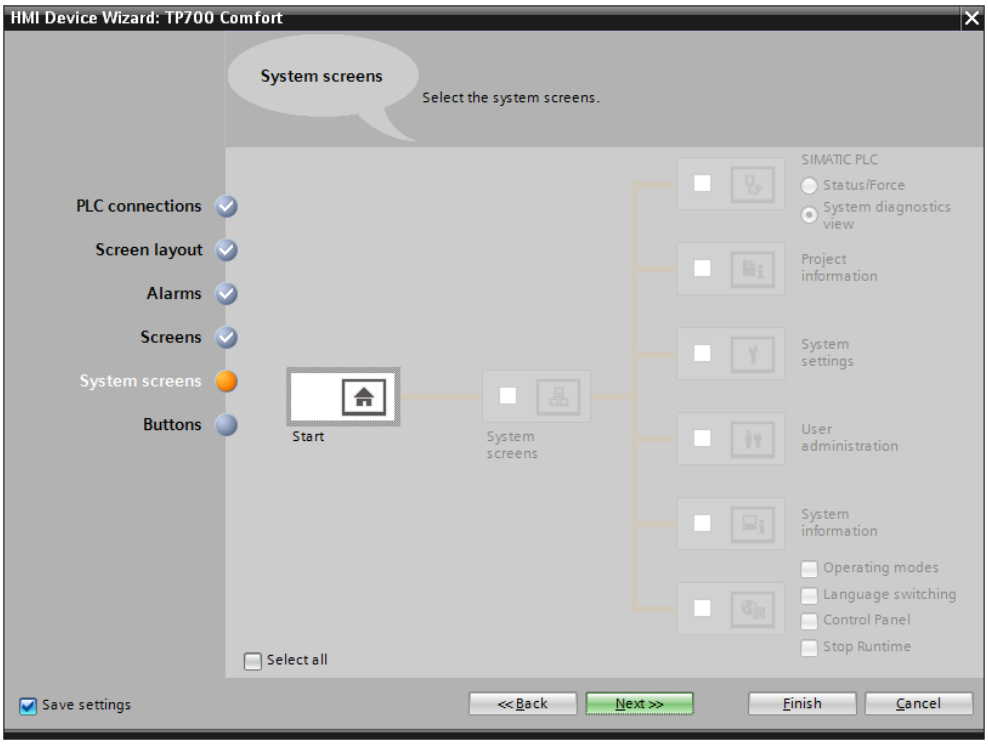
3.


HMI Device Wizard: TP700 Comfort
Alarms
Configure the alarm settings.
PLC connections
Screen layout
Alarms
Screens
System screens
Buttons
Alarms
Unacknowledged alarms
Alarm window
Alarm line top
Alarm line bottom
Pending alarms
Active system events
Preview
Root screen
Save settings
<<Back
Next >>
Finish
Cancel

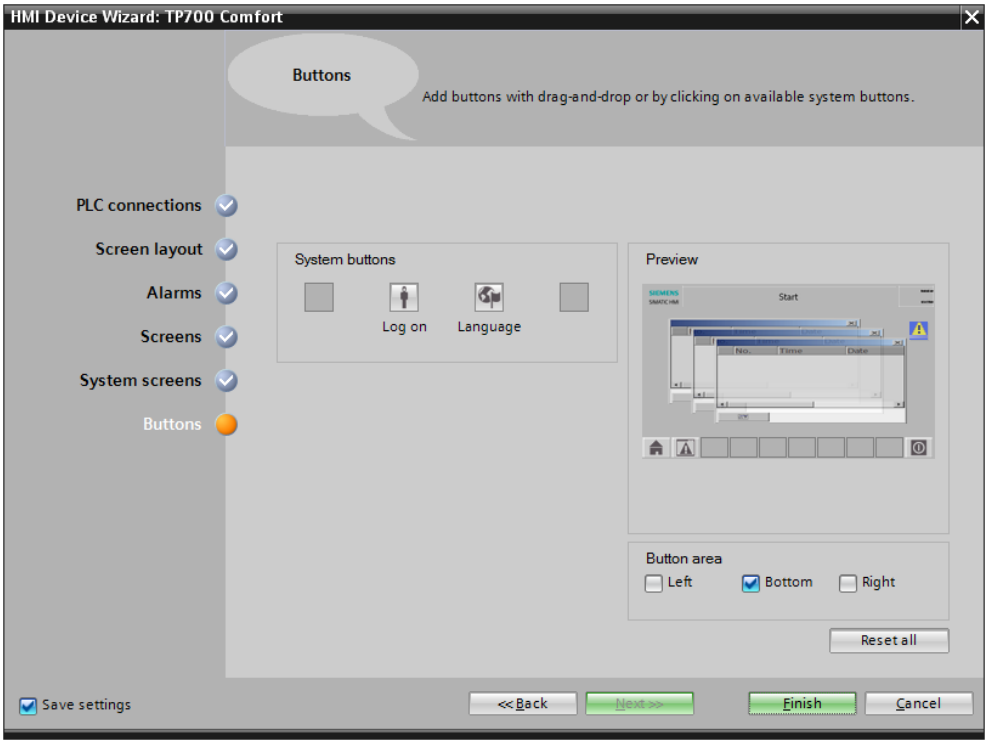
4.


HMI Device Wizard: TP700 Comfort
Screen navigation
Add new screens by clicking this button:
Add screen
Delete screen
Rename
Delete all screens
PLC connections
Screen layout
Alarms
Screens
System screens
Buttons
Start
Alarms
Save settings
<<Back
Next >>
Finish
Cancel

5.


HMI Device Wizard: TP700 Comfort
System screens
Select the system screens.
PLC connections
Screen layout
Alarms
Screens
System screens
Buttons
Start
System screens
SIMATIC PLC
Status/Force
System diagnostics view
Project information
System settings
User administration
System information
Operating modes
Language switching
Control Panel
Stop Runtime
Select all
Save settings
<<Back
Next >>
Finish
Cancel

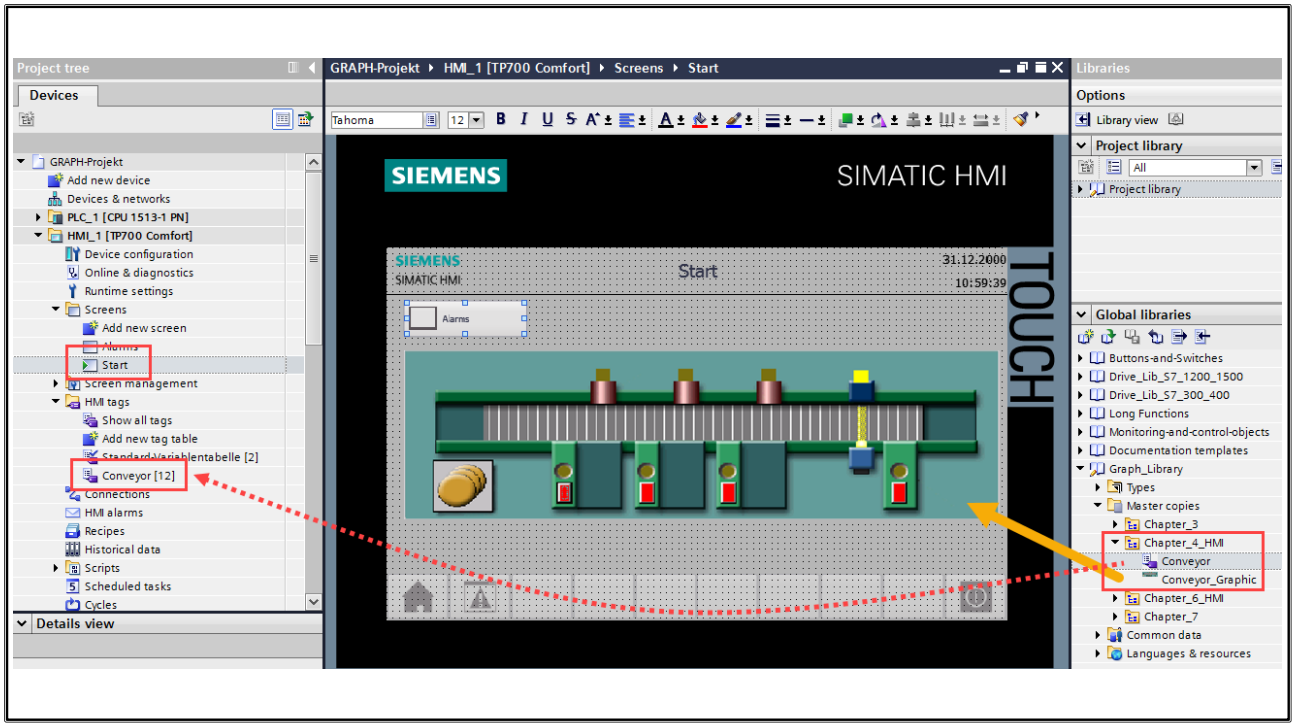
6.



Note

Check whether the HMI device wizard has correctly assigned the IP addresses of the PLC and the TP700. If not, make the necessary corrections (IP address: 192.168.111.xxx Subnetmask: 255.255.255.0).

4.28. Exercise 7: Copying elements from the library



Task

Copy the created tag table and the conveyor image to your project.

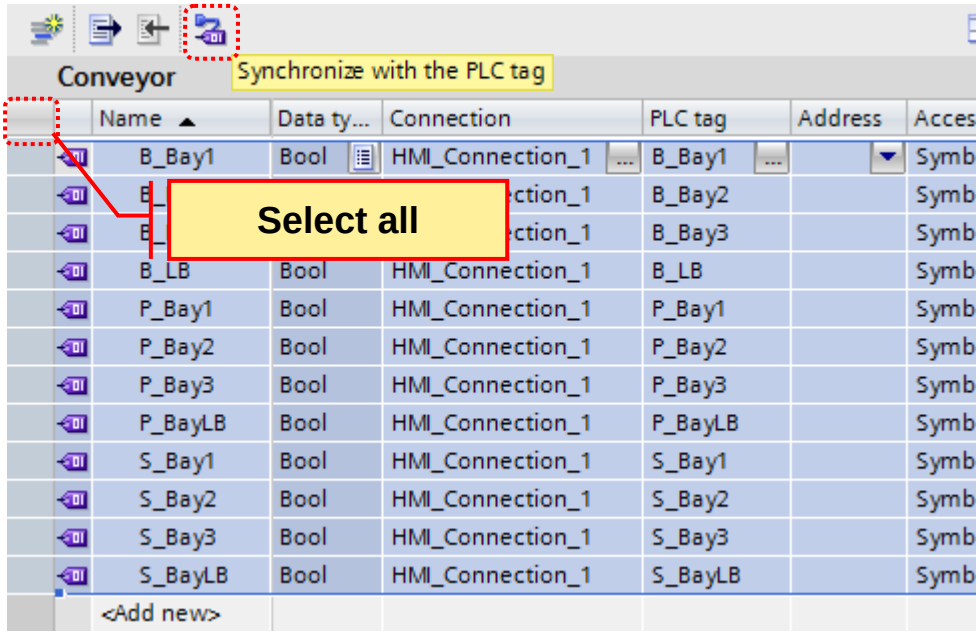
Procedure

1. Use drag-and-drop to move the tag list "Conveyor model" to the "HMI tags" folder.
2. Open the "Start" screen and use drag-and-drop to move the "Conveyor_Graphic" screen to the screen area. Adapt the size of the screen as appropriate.

Note

If the interlinking of the tags from the library in your project does not run smoothly, the following steps must be performed:

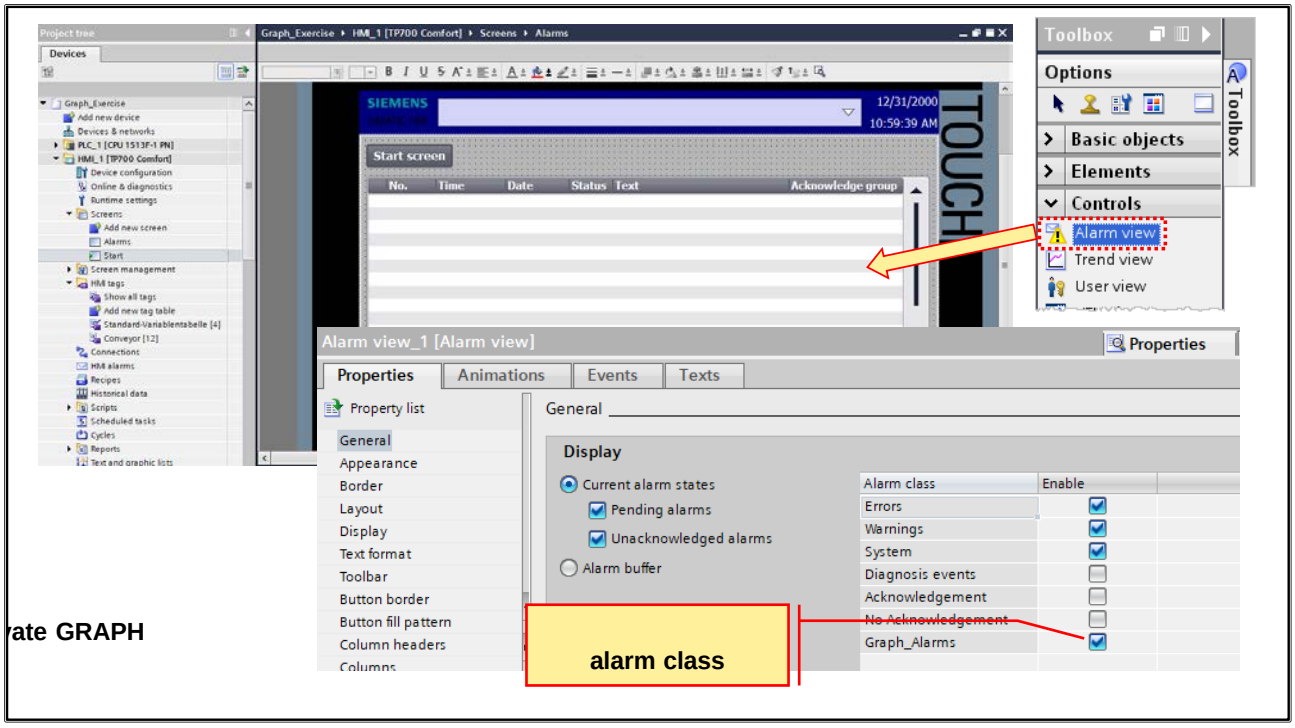
1. Select all HMI tags in the conveyor model HMI tag table and press the "Synchronize with the PLC tag" button.



2. Select the following options and perform the synchronization.

Options for synchronization of WinCC tags
Synchronize PLC tags with WinCC tags if:
Paths of the PLC tags match.
Paths of the PLC tag and the data type match.
Paths of PLC tag, data type and absolute address match.
Data type and absolute address match.
Replace WinCC tag name with PLC tag name
Synchronize
Cancel

4.29. Exercise 8: Configure the alarm view

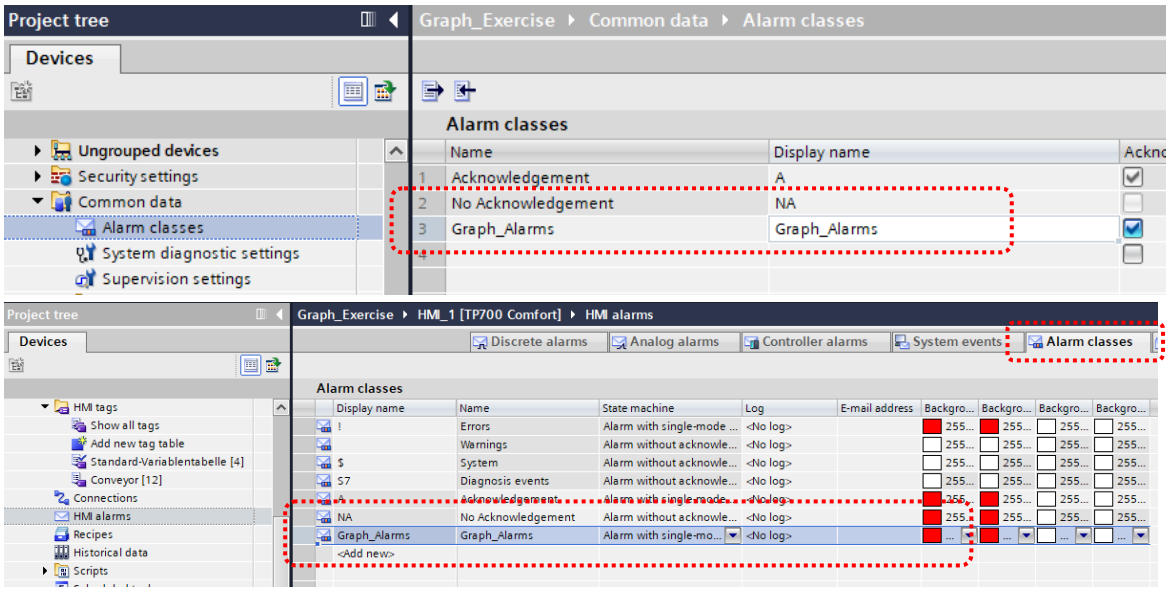


Task

The error messages from the step sequence (interlock and supervision errors) are to be displayed in an alarm view on an HMI touch panel.

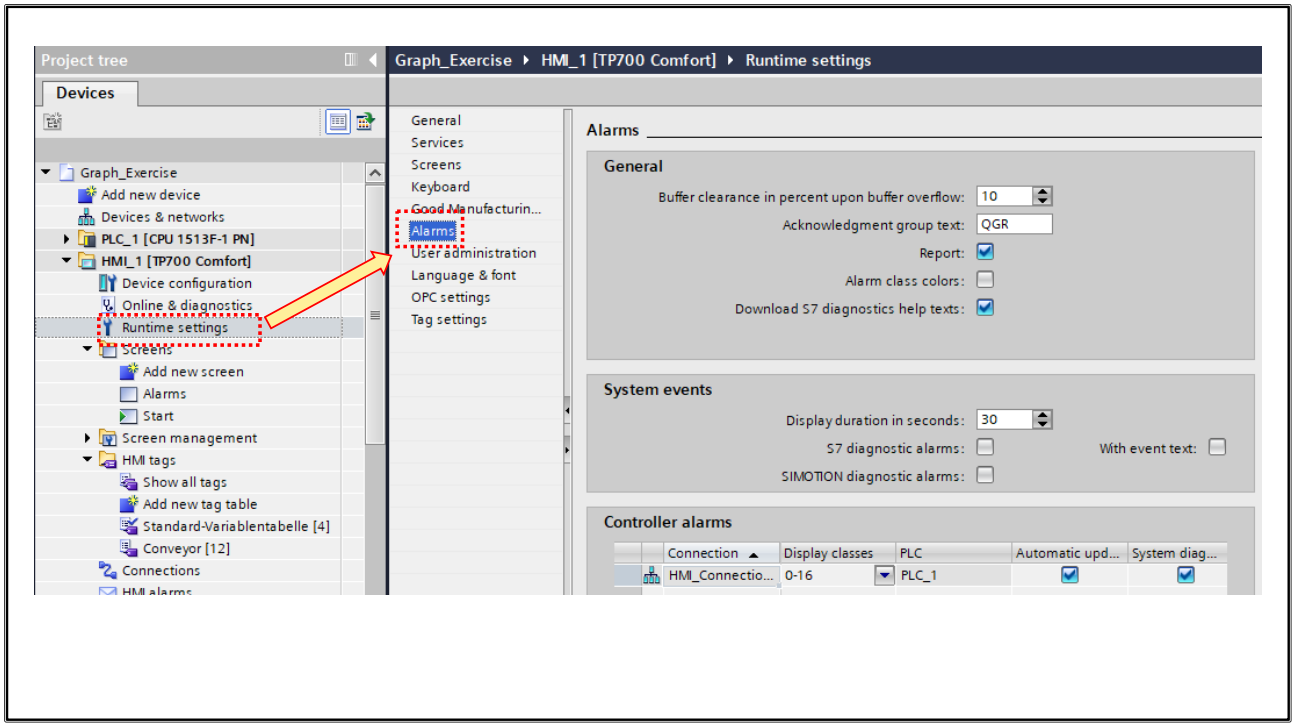
Procedure

1. Check whether the alarm class "GraphAlarmClass_WithAcknowledgement" is included in the HMI alarms:



2. Change to the "Alarms" screen created in the wizard and configure an alarm view like the one shown in the figure there.
3. Adapt the properties of the alarm view as shown.

4.30. Exercise 9: HMI Runtime settings



Task

The Runtime settings of the HMI panel are to be checked and adapted.

Procedure

1. In the Runtime settings of the HMI device, you specify a display duration of the alarm window of 10 s.
2. Compile the HMI project, download the TP700 Comfort and test the alarm view to see whether GRAPH error messages appear.

Table of contents

5

5. Test Functions in GRAPH 5-2

5.1. Handling of the GRAPH programming language 5-3
5.2. Sequence view of a step sequence 5-4
5.3. Options in the sequence view 5-5
5.4. Graphical editing with the mouse 5-6
5.5. Single step view 5-7
5.6. Single step view and actions 5-8
5.7. Online testing of step sequence 5-9
5.8. Sequence control and operating modes 5-11
5.9. Synchronization of step sequence in manual mode 5-12
5.10. Exercise 1: Commissioning with the "Sequence control" function 5-13
5.11. Learning mode for step activation time 5-14

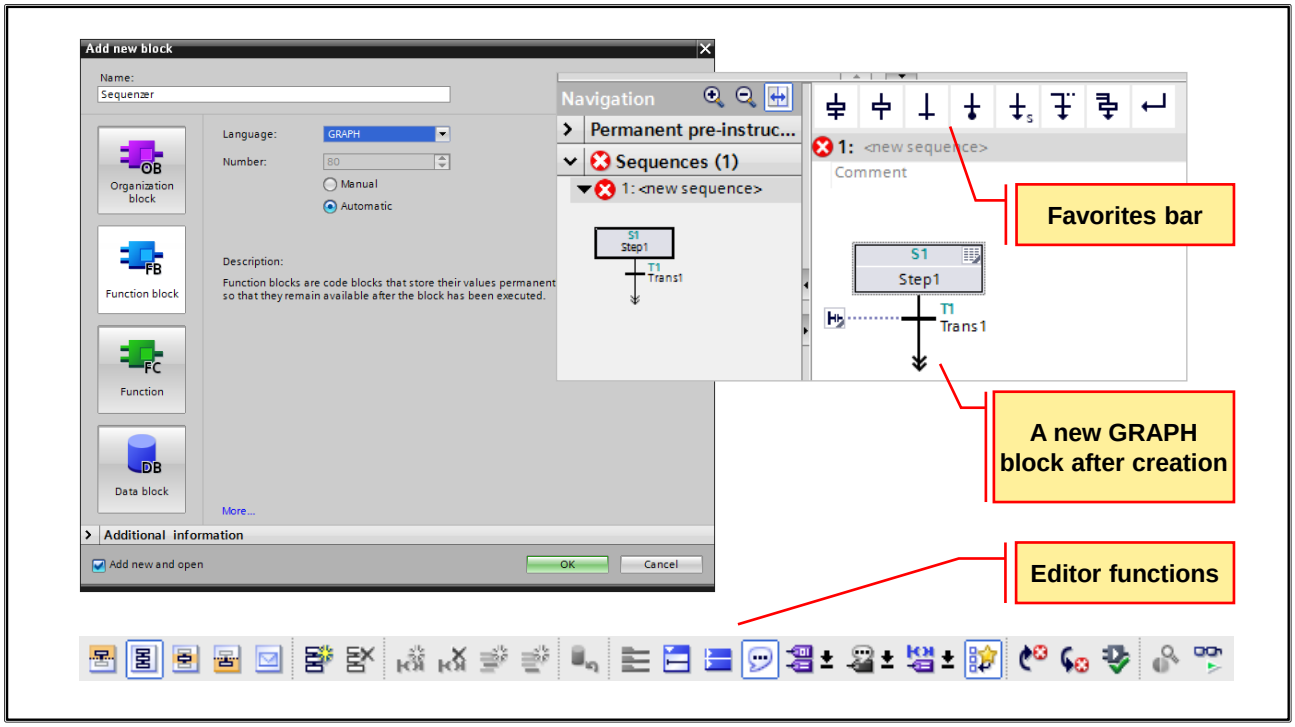
5. Test Functions in GRAPH

The participant should ...

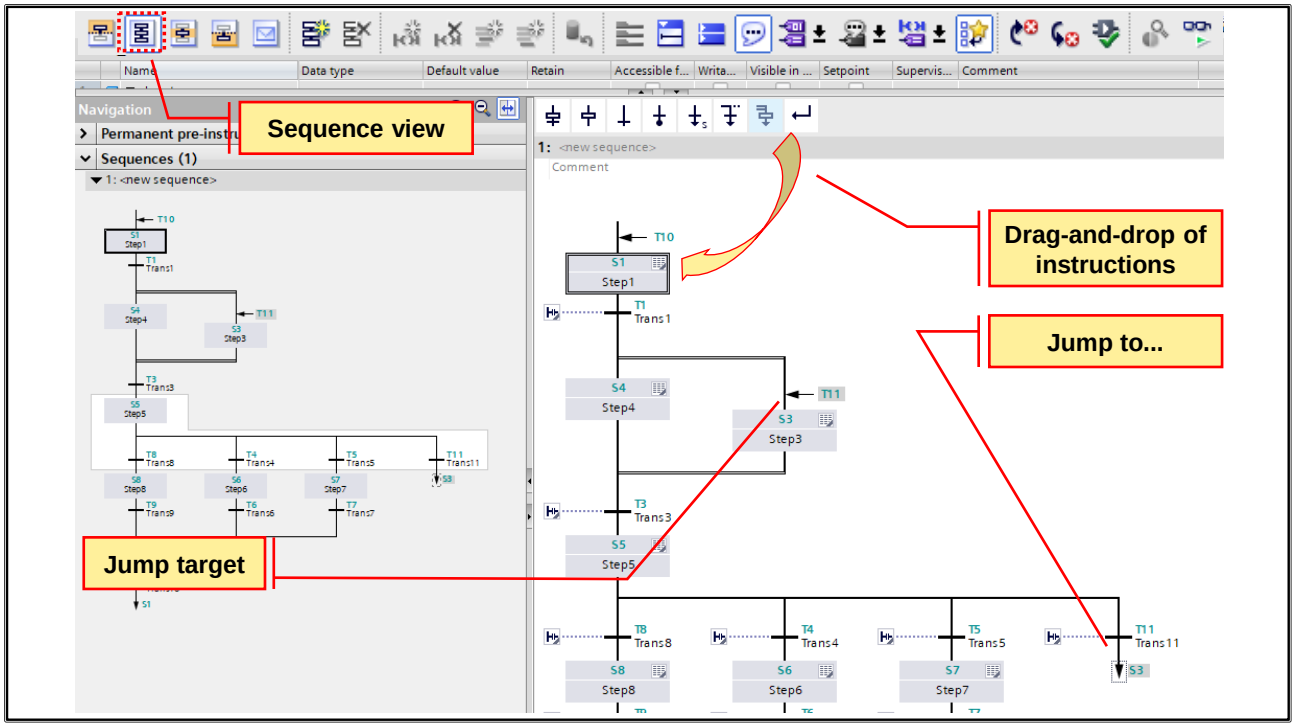
… obtain an overview of the test functions of GRAPH

... become acquainted with the properties of the GRAPH compiler

5.1. Handling of the GRAPH programming language



5.2. Sequence view of a step sequence



Overview

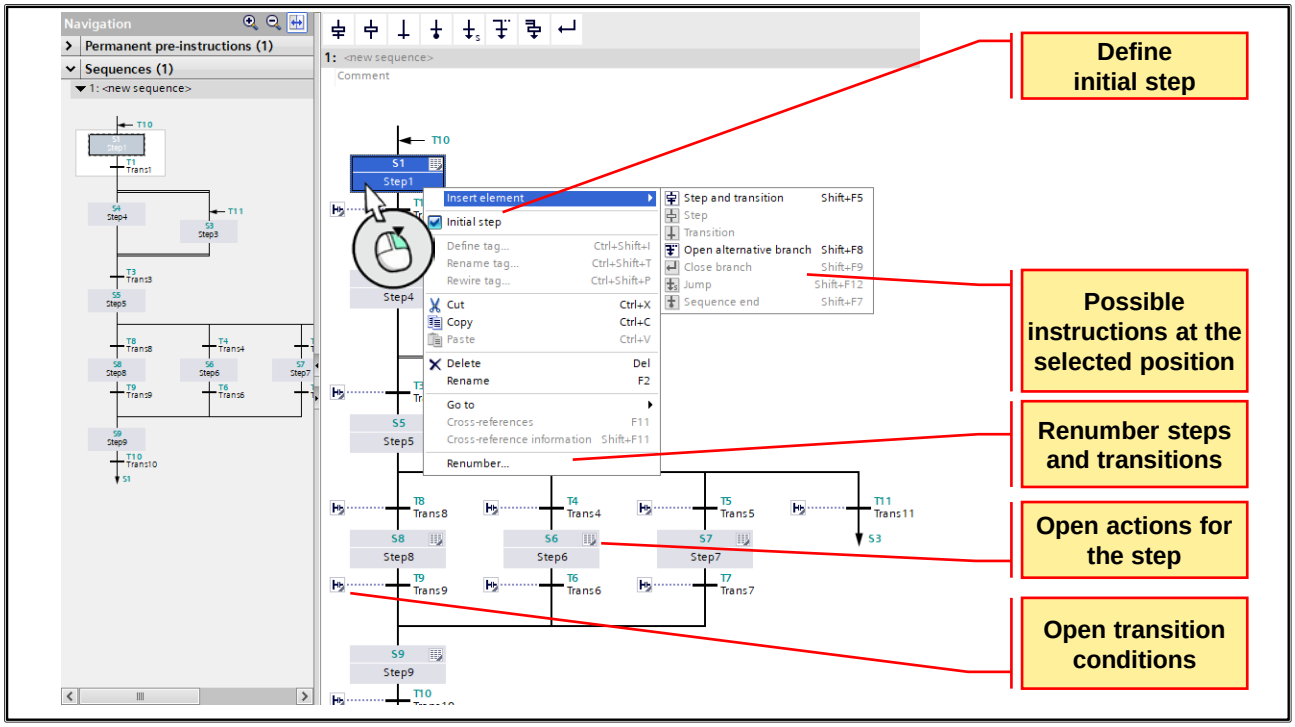
You are supported by graphical programming functions when creating a sequencer. A sequence structure can be easily created without programming expertise by arranging the GRAPH structure elements in such a way that a graphical representation of the sequencer results.

Structure of a sequencer

A sequencer consists of a sequence of steps and transitions. This sequence can be linear or branched. The commands to the plant (actions) are formulated in the steps. The transitions contain the conditions for the advancement from one step to the next. The following symbols represent the elements that a sequencer can be composed of. You can select these symbols from the Favorites bar and move them to the work area using drag-and-drop.

Symbol	Name
	Step and transition
	Step
	Sequence end
	Jump
	Open alternative branch
	Open simultaneous branch
	Close branch

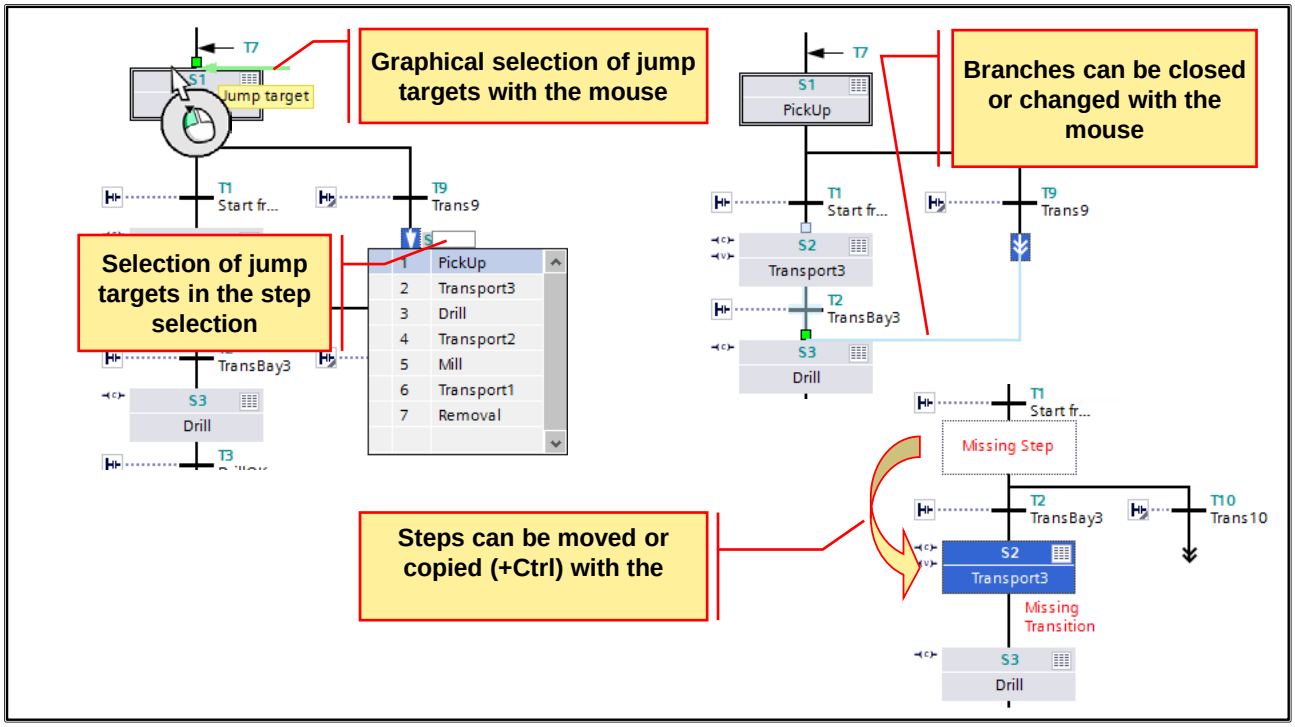
5.3. Options in the sequence view


Define initial step
Possible instructions at the selected position
Renumber steps and transitions
Open actions for the step
Open transition conditions

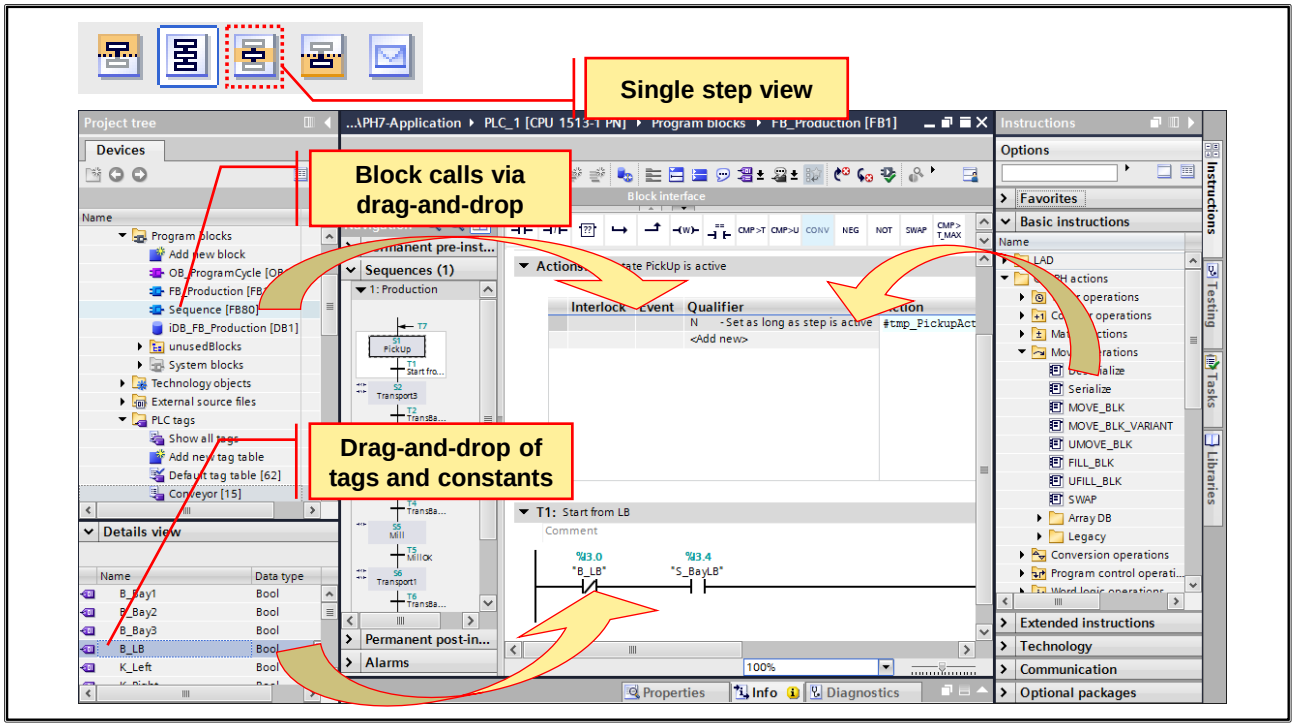
General

The structure of the sequencer is created and refined in the sequence view. The actions of the steps and the transition conditions can also be programmed and checked in this view.

5.4. Graphical editing with the mouse



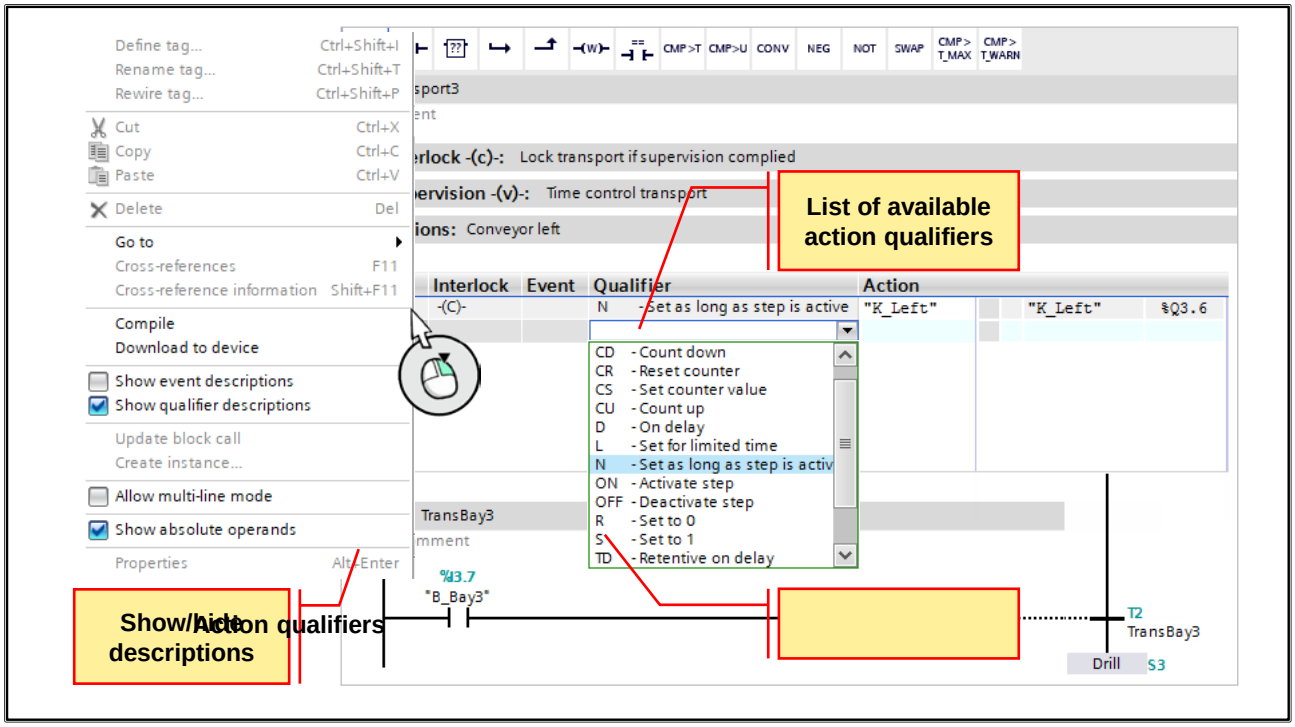
5.5. Single step view



General

As in the sequence view, the actions in the step and the step-enabling conditions of the subsequent transition can be programmed in the single step view. However, the interlock and the supervision conditions of the step can also be programmed here additionally.

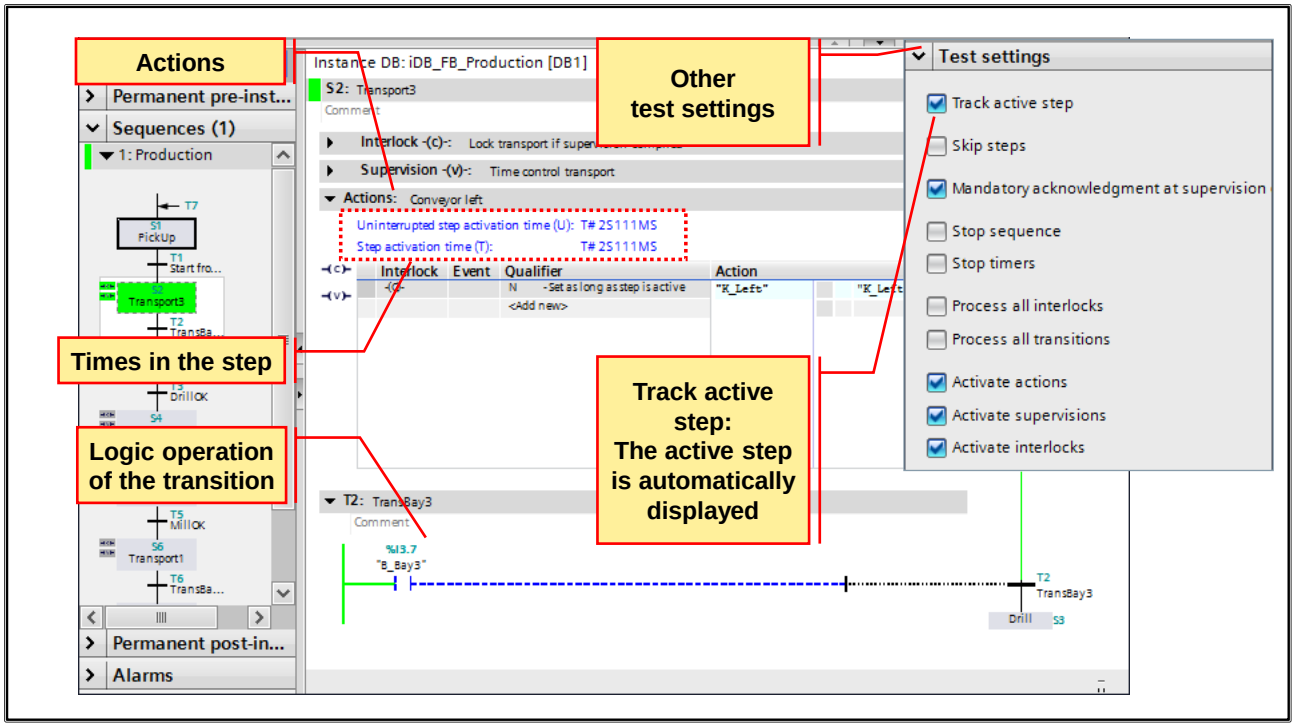
5.6. Single step view and actions



General

The actions belonging to the step can be programmed in both the single step view and the sequence view. The user has access to a ready-made list of actions that he/she can use in the step.
Events and SCL expressions will be covered in later chapters of this course.

5.7. Online testing of step sequence



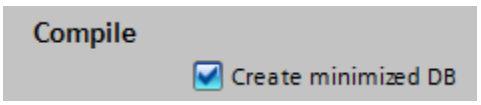
General online test functions

- Display of active steps and steps with faults
- Status display and modify tag

Test settings

On the "Test Settings" tab, various settings can be made for the sequencer to optimally adapt the monitoring of a sequence to the circumstances.

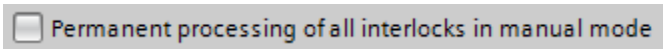
- **Track active step:**
  Displays the currently active step automatically in the single step or sequence view. For a simultaneous branch, the first branch is always displayed.
- **Skip steps:**
  A step whose preceding and subsequent transition are valid will not be activated but instead skipped.
  The default setting of this option is adopted from the corresponding block settings.
  This option is not available when the "Create minimized DB" option is used for the GRAPH function block.



- **Mandatory acknowledgement at supervision error:**
  Specifies whether or not supervision errors have to be acknowledged by the user (see chapter 6 "Self-acknowledgment when transition condition is true").
- **Stop sequence:**
  The sequencer is stopped. It does not advance to the next step even when the transition condition is true.
- **Stop timers:**
  All step activation times are stopped. After resetting this option, the times continue running.

- **Process all interlocks:**
  All interlocks, including those of inactive steps, are processed.
  You can recognize the interlocks that whose condition is true in the sequence view.

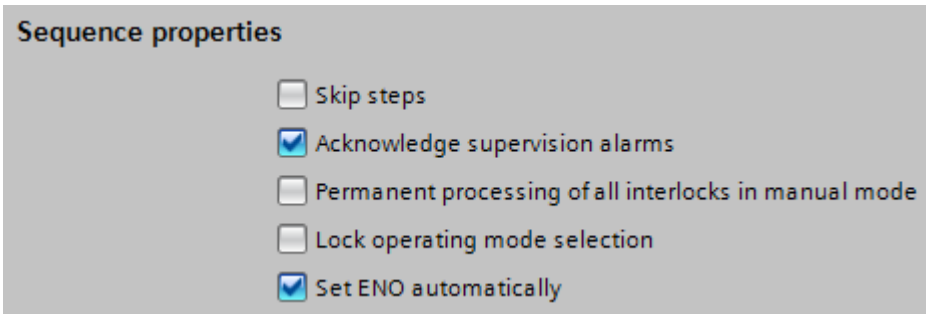
  This option is not available if the "Permanent processing of all interlocks in manual mode" option is selected in the block properties of the GRAPH FB.

- **Process all transitions:**
  All transitions are always processed.
  You can recognize the transitions whose condition is true in the sequence view.
- **Activate actions:**
  This option is selected by default. All actions in the active step are executed if their conditions are true.
- **Activate supervisions:**
  This option is selected by default. If the option is deselected, the conditions in the supervision of the active step are no longer considered. Supervision errors will then no longer be triggered.
- **Activate interlocks:**
  This option is selected by default. If the option is deselected, the conditions in the interlock of the active step are no longer considered. Interlock errors will then no longer be triggered.

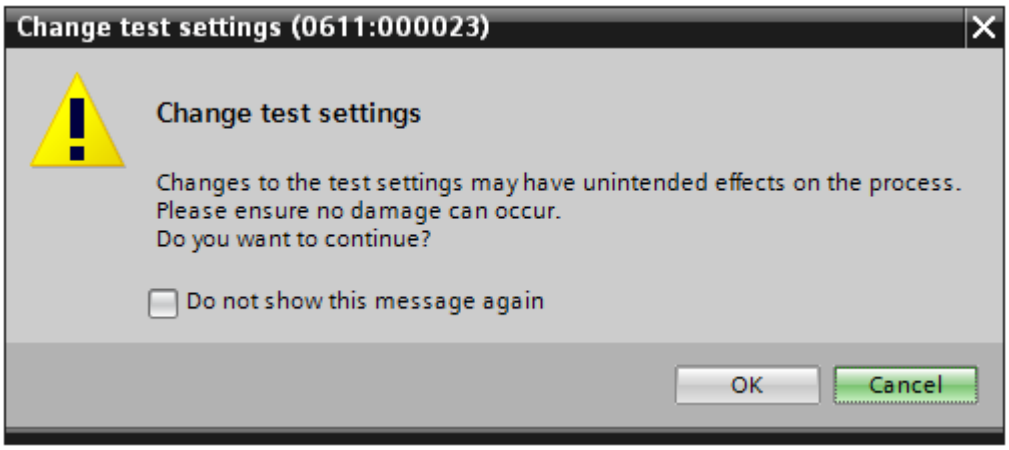
Note

The following attributes can be selected as default settings for a new GRAPH block in the settings of the TIA Portal.

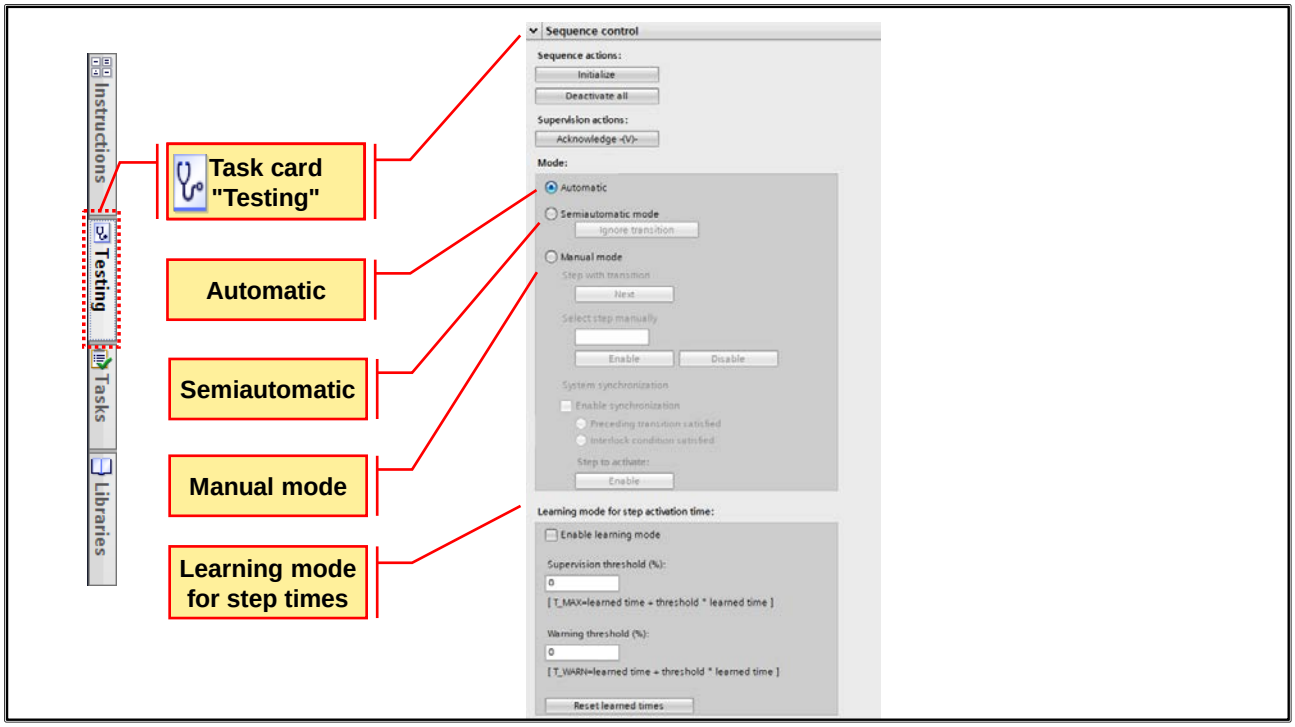


However, these do not have to match the current test settings for the monitoring of the GRAPH FB!

Furthermore, the following note is displayed when a test setting is activated:


Change test settings (0611:000023)
Change test settings
Changes to the test settings may have unintended effects on the process.
Please ensure no damage can occur.
Do you want to continue?
Do not show this message again
OK
Cancel

5.8. Sequence control and operating modes



Sequence control

The "Sequence control" function is a test function that can be used to test the GRAPH sequencer from the TIA Portal in all operating modes. All settings and inputs in the dialog box have an effect like the corresponding FB parameters.

The "Mode" dialog box is used both as an output field that indicates the current settings and as an input field that can be used to change the current states.

Inputs

Inputs in the "Mode" dialog box can be different than the settings you made for compiling the function block. They are higher-ranking than these.

Setting the operating mode

Automatic mode, semiautomatic mode and manual mode can be set. These combination input/output fields indicate the current operating mode. Clicking on one of the options will switch the sequencer to the selected operating mode.

Select step manually

Enter the step number of the step you want to manipulate. Select the action that is to be performed with the step (only possible in "Manual" mode).

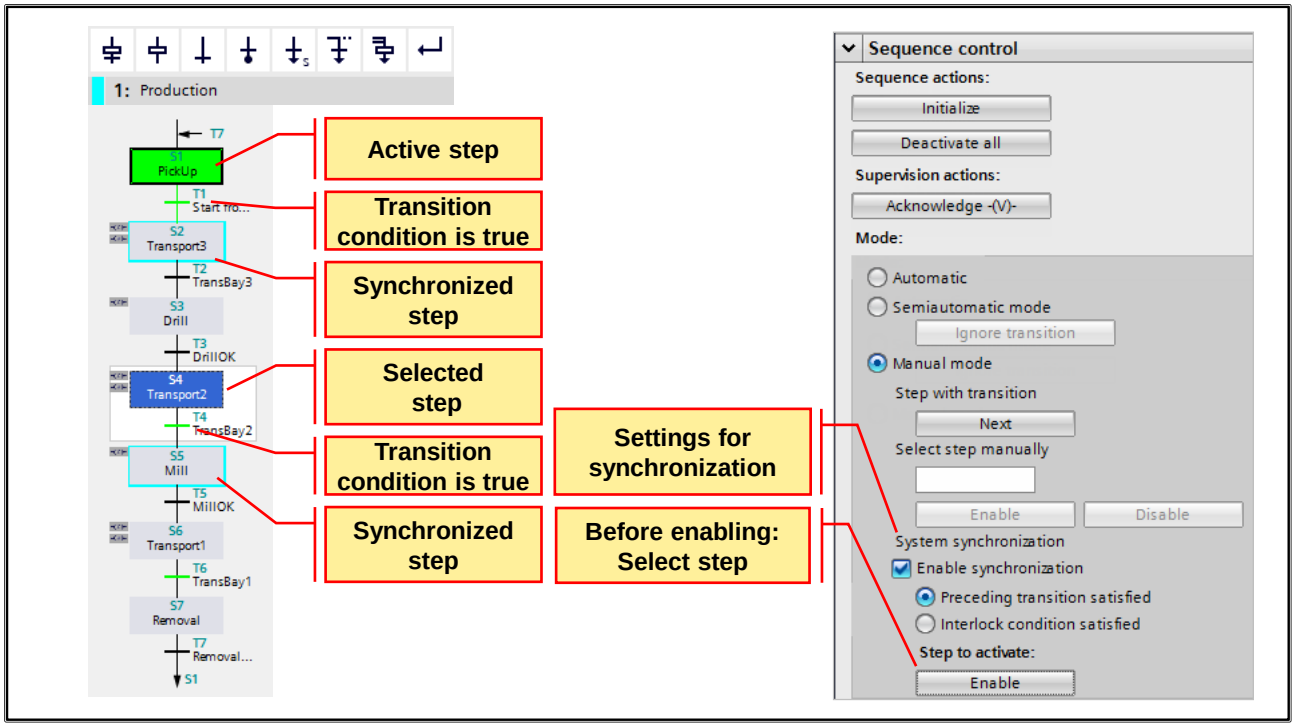
Enable

The selected step is enabled, irrespective of the preceding transition. Actions in the step are executed (default setting).

Disable

The selected step is disabled and thus all actions with "N", "NC" and "Call" are disabled (default setting). The other actions are no longer executed, whereby a "S Q4.0" of course remains active.

5.9. Synchronization of step sequence in manual mode



Synchronization

GRAPH enables the locating of possible synchronization points between the process and sequencer.

If a process is brought manually to another state, it is not longer synchronized. This can be caused, for example by a switchover to manual mode in which any step can be activated.

So that the possibly changed process can run automatically again, the "Synchronization" function enables the locating of potential resynchronization points. GRAPH marks all steps for which the following condition is met:

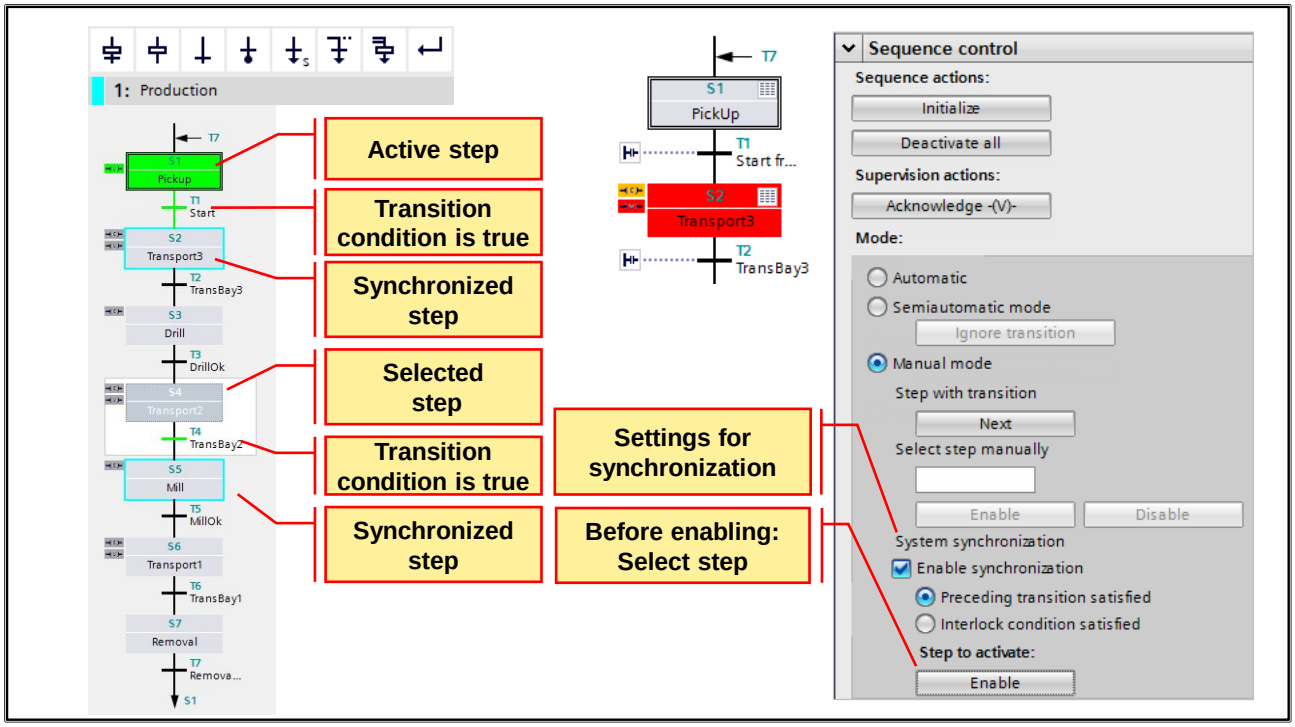
- The condition of the transition preceding the step is true and the condition of the subsequent transition is false.
- The interlock condition of the step is true.

The user is free to switch between these two conditions during runtime.

Requirement

The sequence is in "Manual mode".

5.10. Exercise 1: Commissioning with the "Sequence control" function



Task

Commission the programmed step sequence with the various test settings and sequence control methods.

Procedure

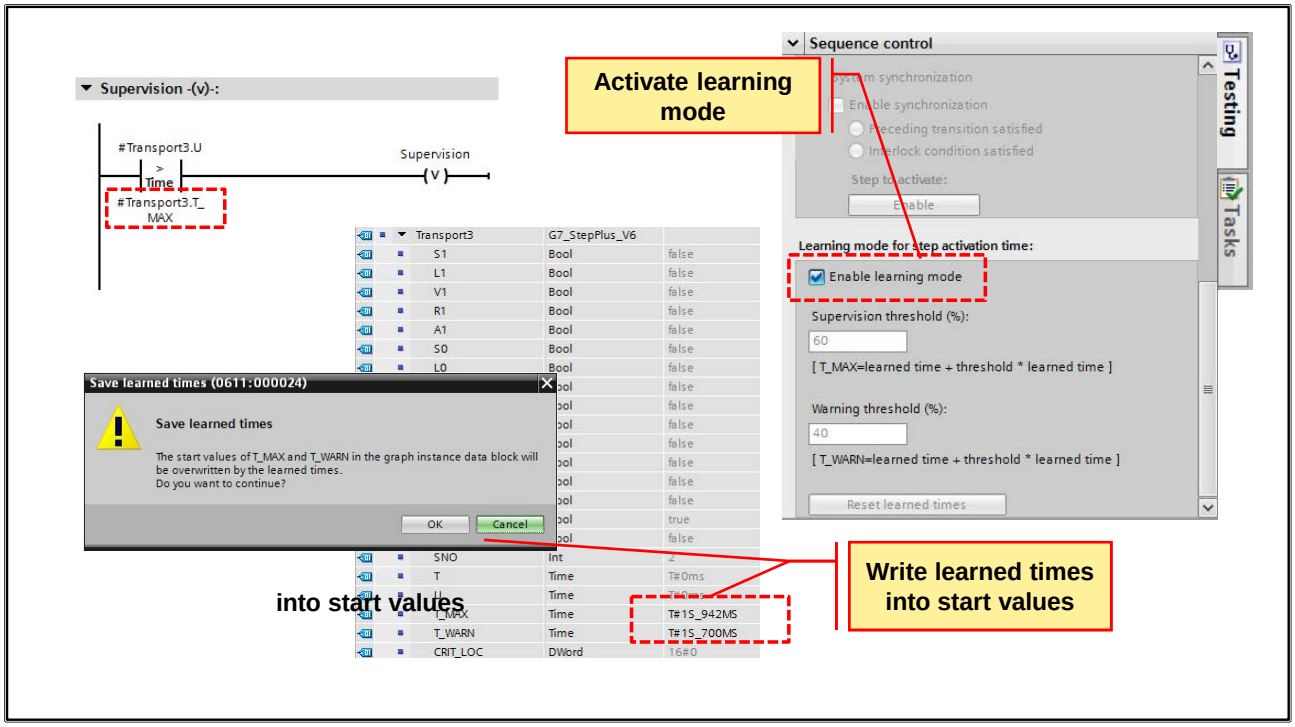
1. Test **all** operating modes of your sequence first with the "Sequence control" function described in this chapter.
2. Move your "assembly line" plant in manual mode to a variety of machine states and use "Synchronization" to display the synchronization points in order to select and activate the right step.
3. Set 'Manual mode' using the "Sequence control" function. Select the step "S5" Mill step and enable it. Then establish a state on the assembly line that is not synchronized with step "S5".
4. Start the "Synchronization" tool and synchronize the sequencer with the plant state of the assembly line.

In most cases, multiple steps are displayed and it is up to the user to pick the right step.

Note

In order to have clear assignment possibilities for synchronization using the GRAPH system, the programmer must take this into consideration when creating the step-enabling conditions.

5.11. Learning mode for step activation time



General

You can use the task card "Testing" to teach in the step activation times. The more frequently the sequence of steps with an active process goes through, the more accurate are the learned times. The user-specified percentage limits are added to the taught-in times and, in total, give the maximum monitoring times for warning and supervision. The times are finally transferred to the start values of the respective Graph Instance DB. The IDB must then be reloaded into the CPU.

Instead of fixed constants, you can now use the step-specific maximum time values from the static parameters in your supervision logics.

Table of contents

6

6. Event-driven Actions 6-2

6.1. Block calls in actions 6-3
6.2. Interlock-dependent actions 6-4
6.3. Actions triggered by an event 6-5
6.4. Example of event-driven actions 6-7
6.5. Event-driven actions in the step 6-8
6.6. Event-driven actions at the step change 6-9
6.7. Events in permanent instructions, transitions, interlocks and supervisions 6-10
6.8. Static parameters of a step (online help) 6-11
6.9. Query of binary sequence data (examples) 6-12
6.10. Query of static signals in the permanent area 6-13
6.11. Transitions with interlock query 6-14
6.12. Exercise 1: Adapt FB Production 6-15
6.13. Use of SCL expressions in GRAPH 6-16
6.14. Running time of single steps 6-18
6.15. Mandatory acknowledgment of errors 6-19
6.16. Acknowledgment in the supervision logic 6-20
6.17. Query of transition states 6-21
6.18. Self-acknowledgment when transition condition is true 6-23
6.19. MOP – Mode of Operation 6-24
6.20. Exercise 2: Integrate automatic and manual modes 6-26
6.21. Exercise 3: Event 6-27
6.22. Exercise 3: Continuation 6-29
6.24. If you want to know more 6-30
6.25. Exercise 4: Self-acknowledgment of supervision errors 6-31

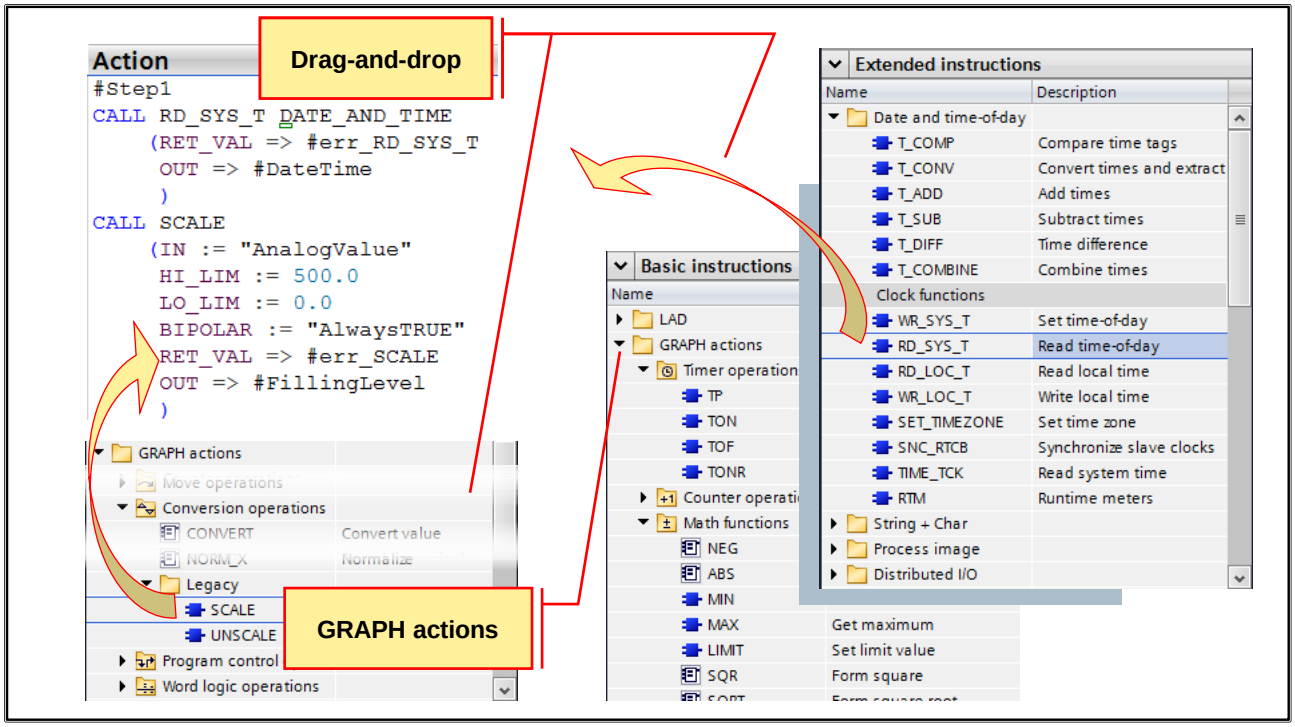
6. Event-driven Actions

The participant should ...

… gain knowledge about event-driven operations and the query of binary chain data

... become acquainted with step interlocks and acknowledgment mechanisms

6.1. Block calls in actions

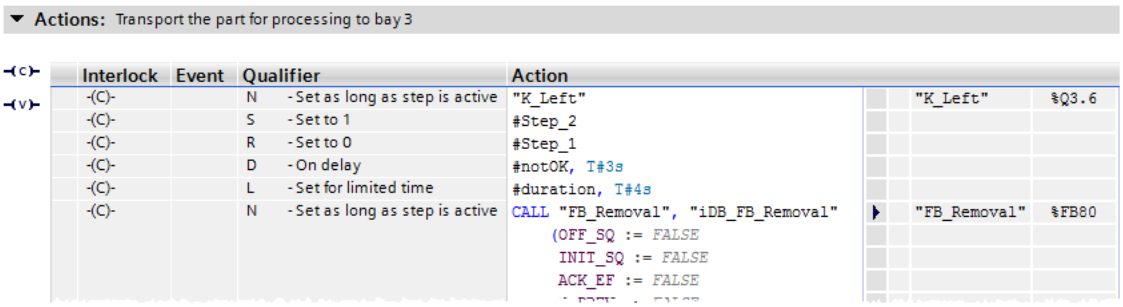


Block calls

Other blocks as well as instructions can be called in a GRAPH block.
This is done simply by moving the desired instruction or block using drag-and-drop. The user then also selects whether or not this call is to be logically combined with an event.
The CALL is then automatically created when the instruction elements are copied to the action area of a step. In the example, the reading of the CPU time is represented by the expanded instruction RD_SYS_T (corresponds to system function SFC1). The SCALE call corresponds to the known conversion block (FC105) for analog values.

6.2. Interlock-dependent actions

Action block with conditional instructions



Conditions:

- An action marked with "C" (Condition) is only executed when the interlock condition (interlock) of the step is true ("C" = 1).
- An interlock error is present when the condition is false ("C" = 0). The conditional actions with "C" are then not executed. The step is marked orange and the error message "Interlock Error" is generated.

Conditions for interlock

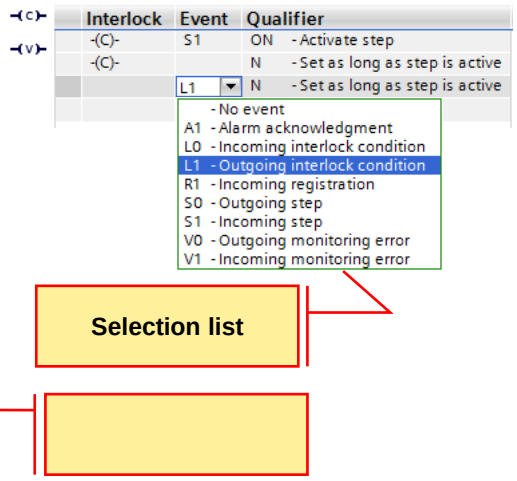
- An action marked with "C" (Condition) is only executed when the interlock condition (interlock) of the step is true ("C" = 1).
- An interlock error is present when the condition is false ("C" = 0). The conditional actions with "C" are then not executed. The step is marked orange and the error message "Interlock Error" is generated.

Operation	Operand	Explanation
NC	%Q1.0	As long as the step is active and the condition is true, signal %Q1.0 = 1, otherwise 0.
SC	%Q1.0	As long as the step is active and the condition is true the first time, %Q1.0 is set to 1 and remains set to 1 afterwards.
RC	%Q1.0	As long as the step is active and the condition is true the first time, %Q1.0 is set to 0 and remains set to 0 afterwards.
DC	%Q1.0 T#<const>	After the end of the time specified in <const> and as long as the step is active and the condition is true, signal %Q1.0 = 1. If the step is not active, the signal = 0.
LC	%Q1.0 T#<const>	As long as the step is active and the condition is true, signal %Q1.0 = 1 for specified time <const>. If the step is not active, the signal = 0.
CALLC	FB5,DB3	Calls the specified block when the condition is true. After execution of the block, the GRAPH program continues running.

6.3. Actions triggered by an event

Event-driven instructions cause a one-time execution of the action in the cycle in which the event occurs.

A1 = An alarm is acknowledged
R1 = A registration occurs (REG_S, REG_EF, max. interface)
L1 = Interlock condition is false, i.e. fault occurs (signal state = "0")
L0 = Interlock condition is true, i.e. fault is rectified (signal state = "1")
V1 = Supervision condition is true, i.e. fault occurs (signal state = "1")
V0 = Supervision condition is no longer true, i.e. fault is rectified (signal state="0")
S1 = Step is activated
S0 = Step is deactivated


Interlock
Event
Qualifier
-(C)- S1 ON - Activate step
-(C)- N - Set as long as step is active
L1 N - Set as long as step is active
- No event
A1 - Alarm acknowledgment
L0 - Incoming interlock condition
L1 - Outgoing interlock condition
R1 - Incoming registration
S0 - Outgoing step
S1 - Incoming step
V0 - Outgoing monitoring error
V1 - Incoming monitoring error
Selection list
"Event qualifier"

Overview

An event can be detected and logically combined with an action. This allows you not only to monitor single steps but also to monitor and influence the entire sequential control.

An "Event qualifier" is a one-time signal that is generated by the GRAPH system on entry into or exit from a step, on occurrence of the acknowledgment or the rectification of a fault (interlock or supervision), thereby enabling the user to logically combine an action with this event.

The list of possible qualifiers in GRAPH (L1, L0, V1, V0, S1, S0, A1, R1) is presented in the slide with a brief explanation.

ON / OFF

The **ON** and **OFF** operations can be used to activate or deactivate other single steps "S(i)" of a sequence; "**S_ALL"** can be used to activate or deactivate all other steps.

Event	Operation	Operand	Explanation
S1 S0 V1 V0 L1 L0 A1 R1	 ON OFF	Si i = Step number	Activate or deactivate step dependent on event
L1 S1 V1	OFF	S_ALL	Deactivate all steps dependent on the event, except for the step containing the action.

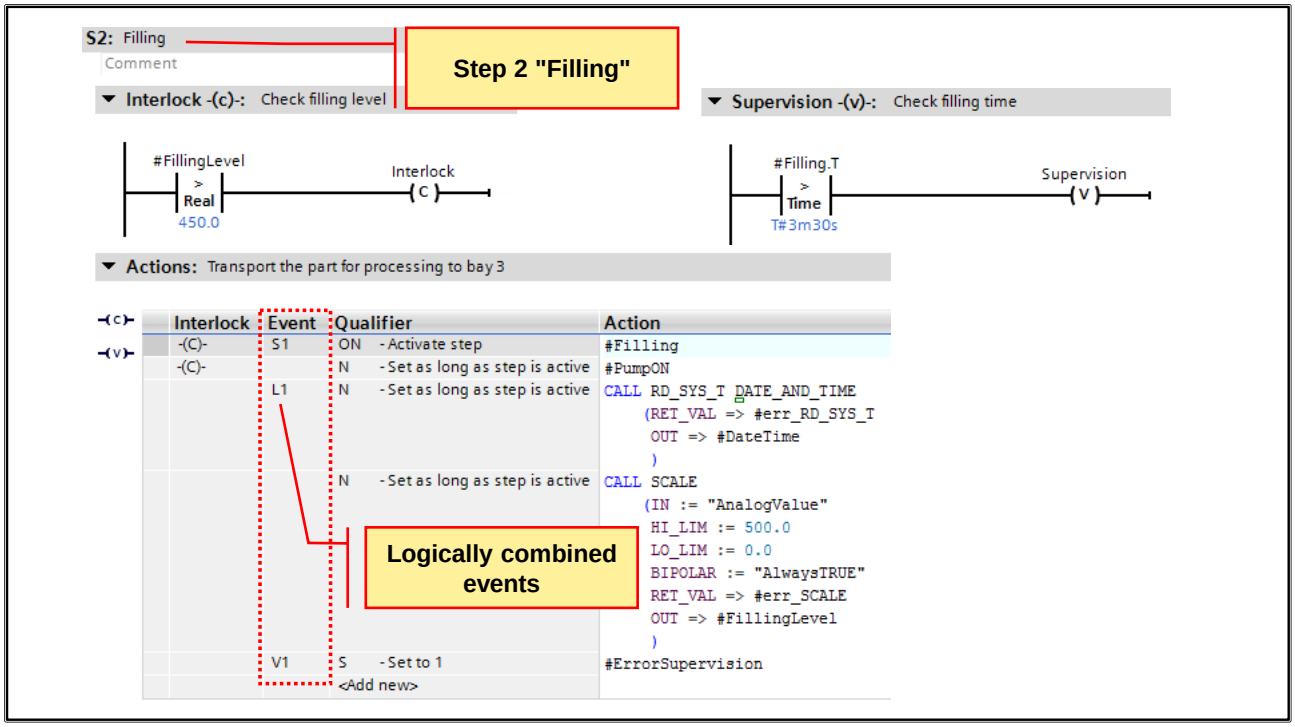
Note

The **ON** and **OFF** operations have an effect only at the end of the action list. That is, it makes no difference which order these appear in the action list. All other actions are executed on a time-dependent basis beforehand. Even events such as "S1" do not affect this time sequence.

The **ON** operation triggers activation of the specified step while simultaneously triggering deactivation of the step in which this was selectively executed.

The **Off** operation can also deactivate the step in which this was executed.

6.4. Example of event-driven actions



Example

The single step view of step S2 ("Filling") in the figure above contains static and event-driven actions.

Static actions:

- Control of a pump with interlock condition **(NC)**.
- The SCALE scaling block reads a fill level (analog value) and scales this to a measuring range of 500 liters (**CALL**).
- When the limit of 450 liters is exceeded, the interlock switches off the pump. The fill level is available at an output parameter of the GRAPH FB.

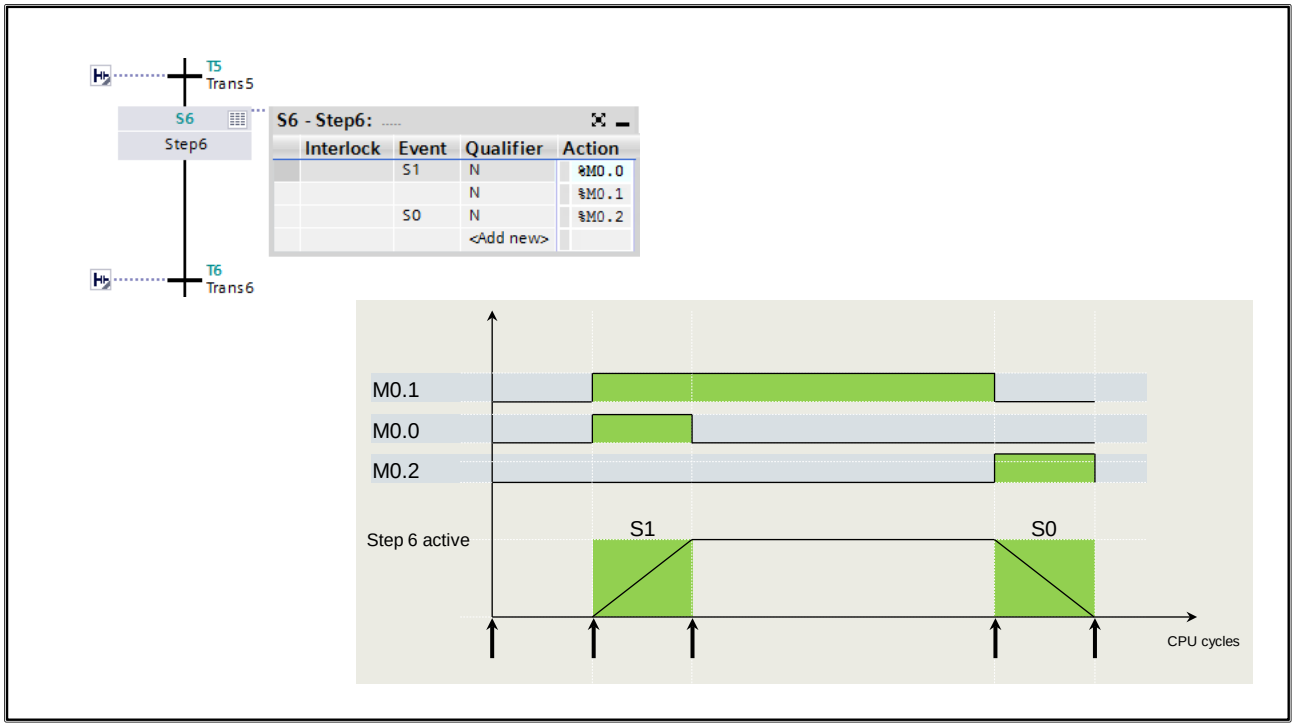
Event-driven actions:

- Upon occurrence in step 2 **(S1)**, step 1 of the same sequence complex is activated but only when the interlock condition for step 2 is true **(ON C)**.
- Upon occurrence of an interlock fault in step 2, the RD_SYS_T function (SFC1) is called once **(L1 CALL)** in order to read out the date and time from the CPU and store them in a static tag. The parameter assignment takes place in the operand part of the action.
- Upon occurrence of a supervision fault (time monitoring for T#3Min30Sec), the static tag "Error_Supervision" is latched **(V1 S)**.

Note

In the example, user-defined tags, e.g. #FillingLevel and #DateTime, are used as temporary declarations in the step sequence block.

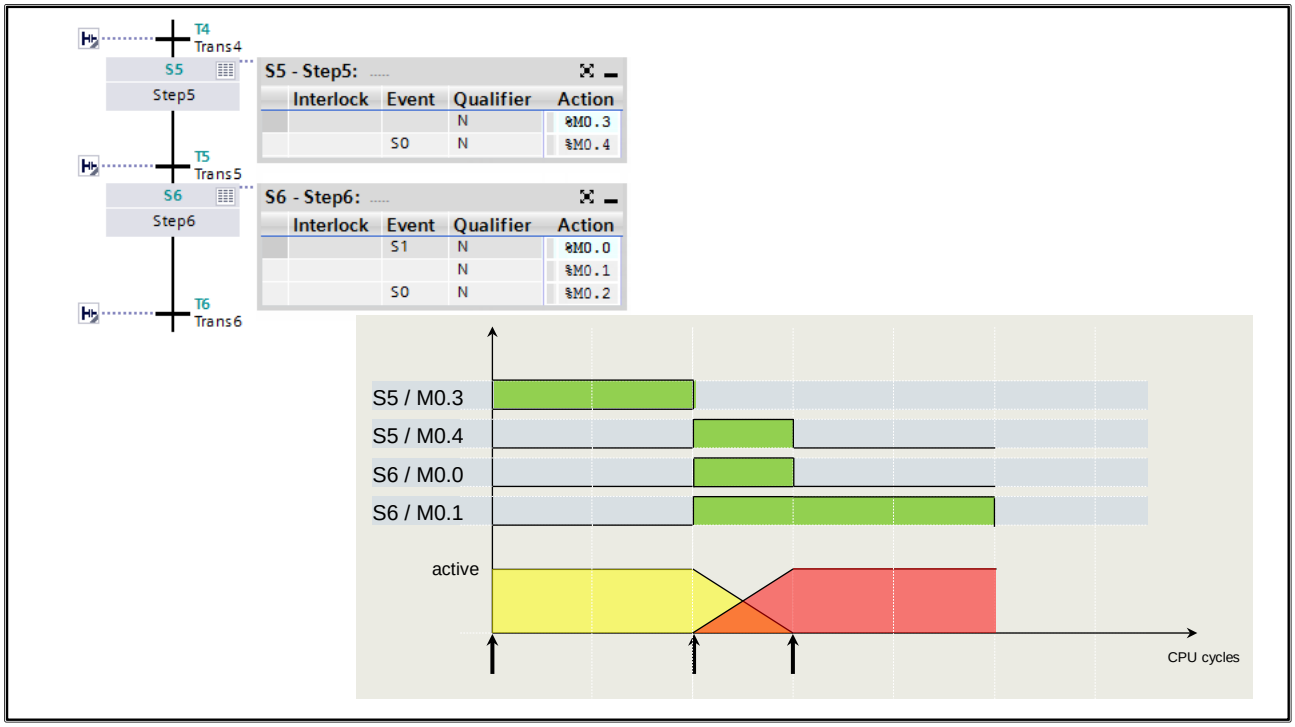
6.5. Event-driven actions in the step



Time lapse of actions in the step

In step "S6", bit memory %M0.2, %M0.3 and %M0.4 are provided with qualifier "N – Set as long as step is active". In addition, %M0.2 is also logically combined with the event "S1 – Incoming step" and %M0.4 with the event "S0 – Outgoing step".
It can be seen from the diagram that actions with qualifier "N" are set when the step becomes active. %M0.2 and %M0.3 are thus set upon activation of step "S6" whereby %M0.2 resulting from event "S1" is TRUE only for one cycle.
As soon as the step is exited, all actions with qualifier "N" are immediately set to FALSE. Actions with qualifier "N" that were logically combined with the event "S0" are TRUE for one cycle. Thus, %M0.4 in the example.

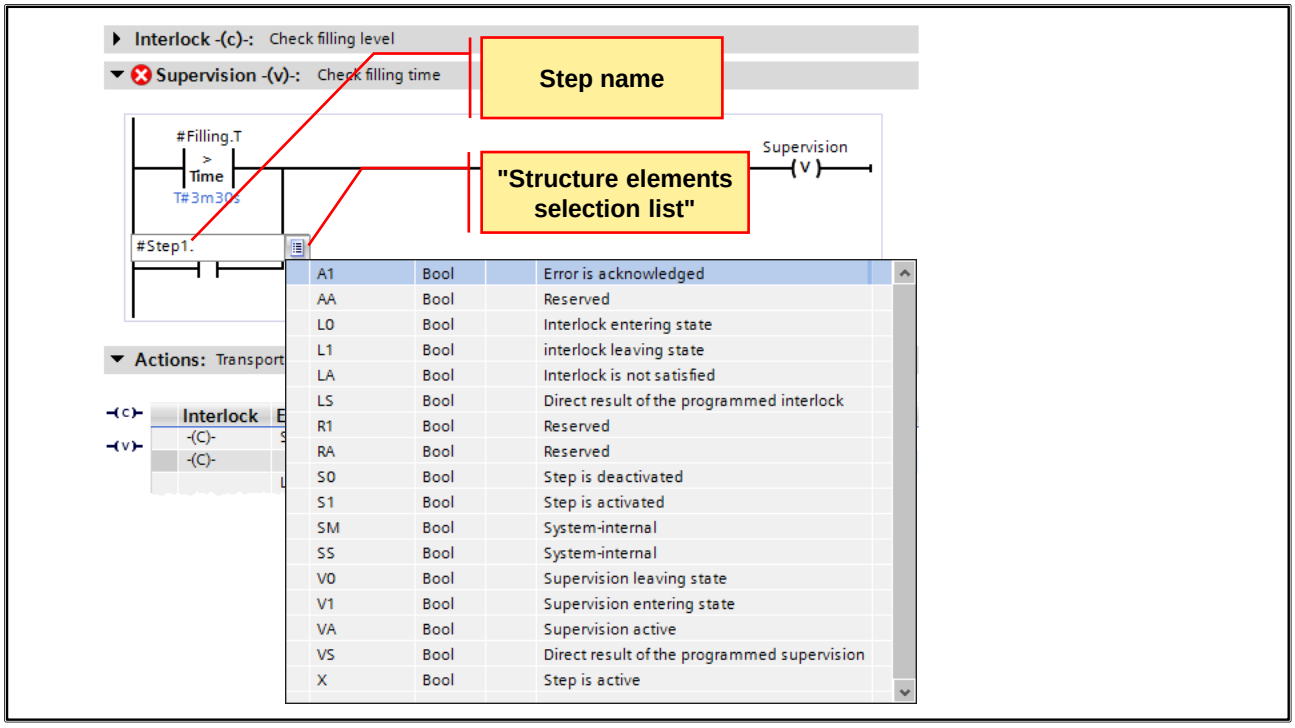
6.6. Event-driven actions at the step change



Time lapse of actions at the step change

Because actions with qualifier "N" are set when a step is activated while actions with qualifier "N" and the event "S0" are set for exactly one cycle when a step is exited, the situation can occur in which actions from the just exited step and actions from the activated step are active simultaneously for one cycle.
In the above example, when step "S5" is exited, %M0.1 is set to TRUE, but when step "S6" is activated, %M0.2 and %M0.3 are also set to TRUE. These all have the value TRUE for one cycle. Afterwards, %M0.1 and %M0.2 are set to FALSE and only %M0.3 remains set to TRUE until step "S6" is deactivated.

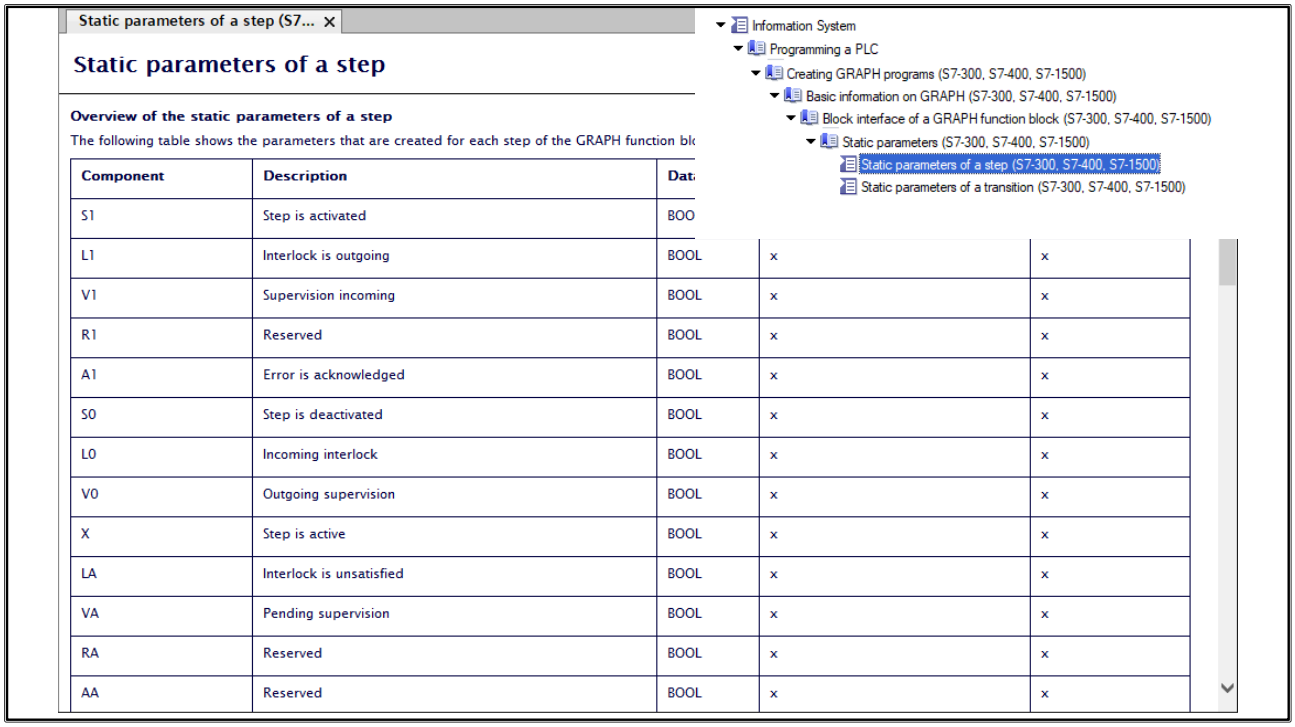
6.7. Events in permanent instructions, transitions, interlocks and supervisions



Overview

Besides the logic operation with actions, events can also be used in permanent instructions, transitions, interlocks and supervisions.
There are more selection options here compared to the logic operation with an action. All possible event qualifiers are shown on the slide. These are used by first writing the step name followed by the qualifier.

6.8. Static parameters of a step (online help)



Step structure

Every step in the sequencer of GRAPH has internal static parameters that can be used for controlling the sequencer and for signaling states. A detailed overview of all parameters can be found in the online help of the TIA Portal.

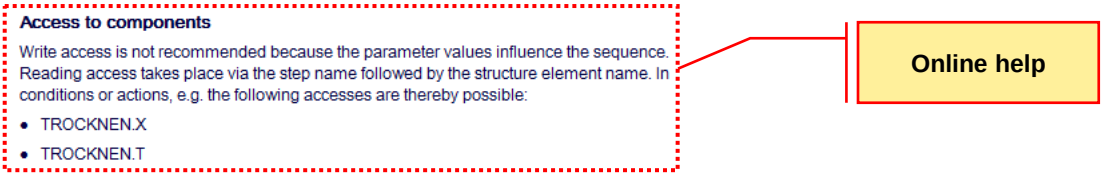
6.9. Query of binary sequence data (examples)

Examples of static signals (#Check part = Step 4)

- #Check part.X = Step no. i is active
- #Check part.LA = Interlock is unsatisfied
- #Check part.VA = Pending supervision

One-time signals = Pulses of the edge evaluation

- #Check part.S1 / S0 = Step becomes active / inactive
- #Check part.L1 / L0 = Incoming / outgoing interlock fault in the step
- #Check part.V1 / V0 = Incoming / outgoing supervision in the step
- #Check part.A1 = Fault in step is acknowledged


Access to components
Write access is not recommended because the parameter values influence the sequence. Reading access takes place via the step name followed by the structure element name. In conditions or actions, e.g. the following accesses are thereby possible:
TROCKNEN.X
TROCKNEN.T
Online help

Overview

The option exists in GRAPH to use certain information about steps as operands in transitions, permanent conditions, supervisions and interlocks.

One-time signals

The "qualifiers" listed in the first section are used for one-time activation of functions. These one-time signals correspond to a STEP 7 edge evaluation.

An event is the change of the signal state, for example, of a step, a step interlock, or a step supervision, the acknowledgement of an alarm or an incoming registration.

Static signals

The static signals are used to query step states in sequencers. All evaluated information is stored in the instance DB.

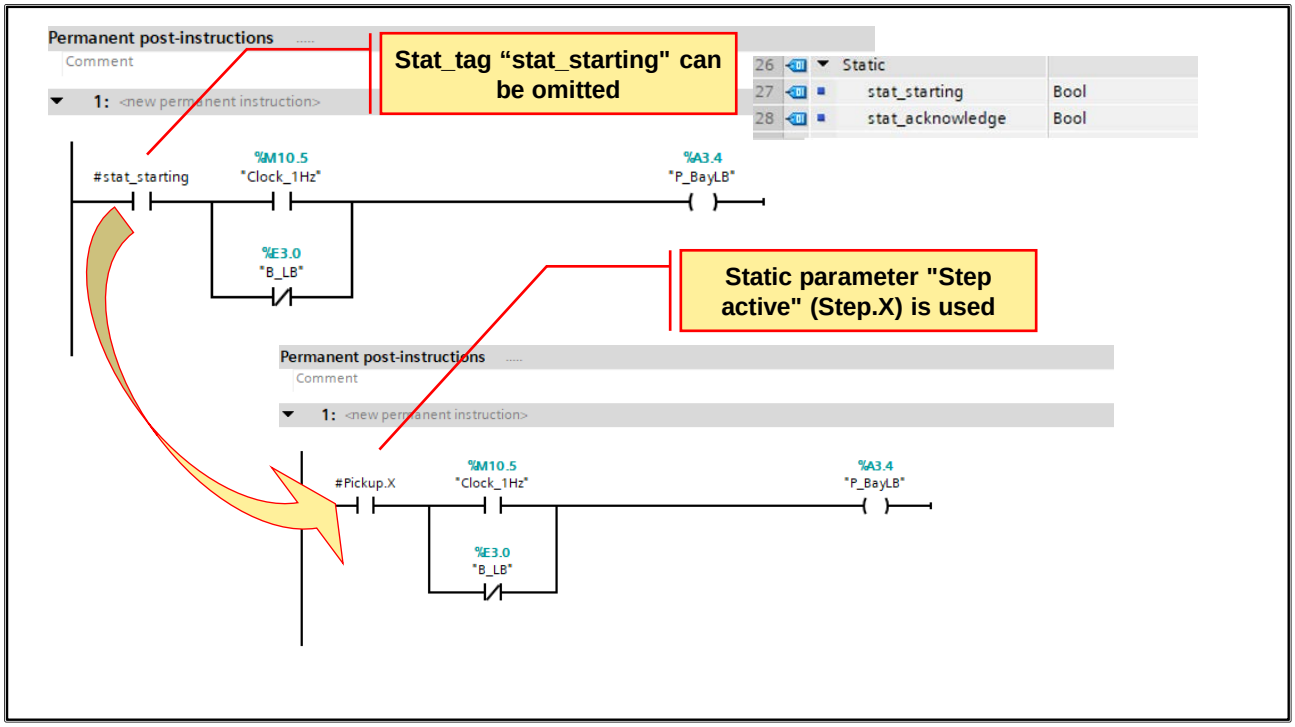
The first letter after the step name has the following meaning:

- **L**: Incoming or outgoing inter**L**ock
- **V**: Incoming or outgoing super**V**ision
- **A**: **A**cknowledge
- **S**: **S**tep becomes active or inactive
- **X**: Step **is active**

The second letter means:

- **0**: Deactivation or **outgoing** fault
- **1**: Activation of **incoming** step
- **A**: Interlock fault or supervision error is **A**ctive

6.10. Query of static signals in the permanent area



Static signals

Static signals can be used in all applications in which general logic operations (without interlock) are needed, such as

- Lower-level setup mode (step-independent manual mode).
- Interlocks that are effective independent of steps.
- Flashing lights that are controlled by clock memory (see figure).

The corresponding logic is placed in a separate block or in the permanent part of the sequence complex. In both cases, the option exists to integrate step activities, acknowledgments, supervision signals, etc. in logic operations.

Permanent part

The advantages of integrating these types of logic operations in the permanent part of the sequence are as follows:

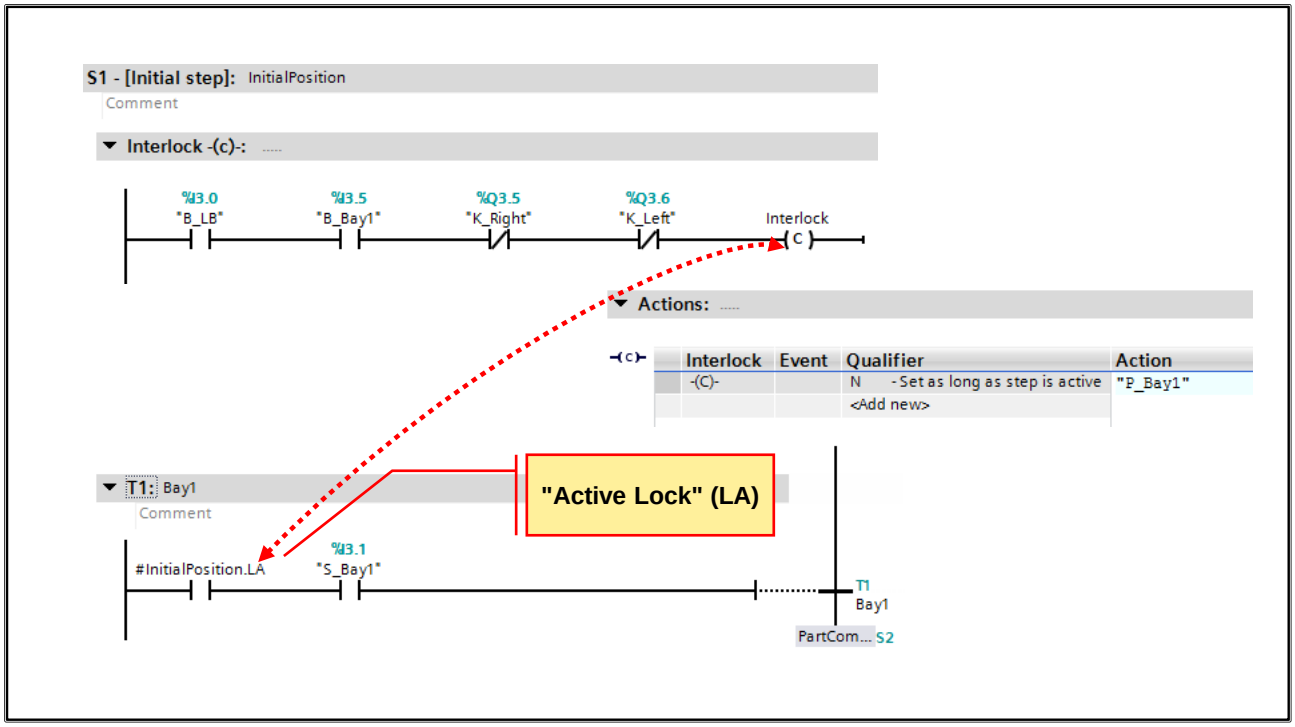
- Simplified access to sequence data via qualifiers is possible.
- Only one block (the GRAPH block) has to be managed and maintained.

Example

In the example shown, the static signal "**X**" is used for the following:

- To activate the indicator light "P_BayLB" with flashing light while step "PickUp" (#PickUp.X) is active.
- To activate the indicator light "P_BayLB" with steady light when step "PickUp" is active and light barrier "B_LB" is occupied.

6.11. Transitions with interlock query



Advancing

If an interlock fault occurs in a step, when a step enabling condition is true, the next step is activated nevertheless.

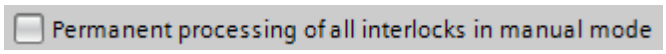
The above example shows a possibility of preventing this advancement when the conditions for the initial position are not present.

Lock active

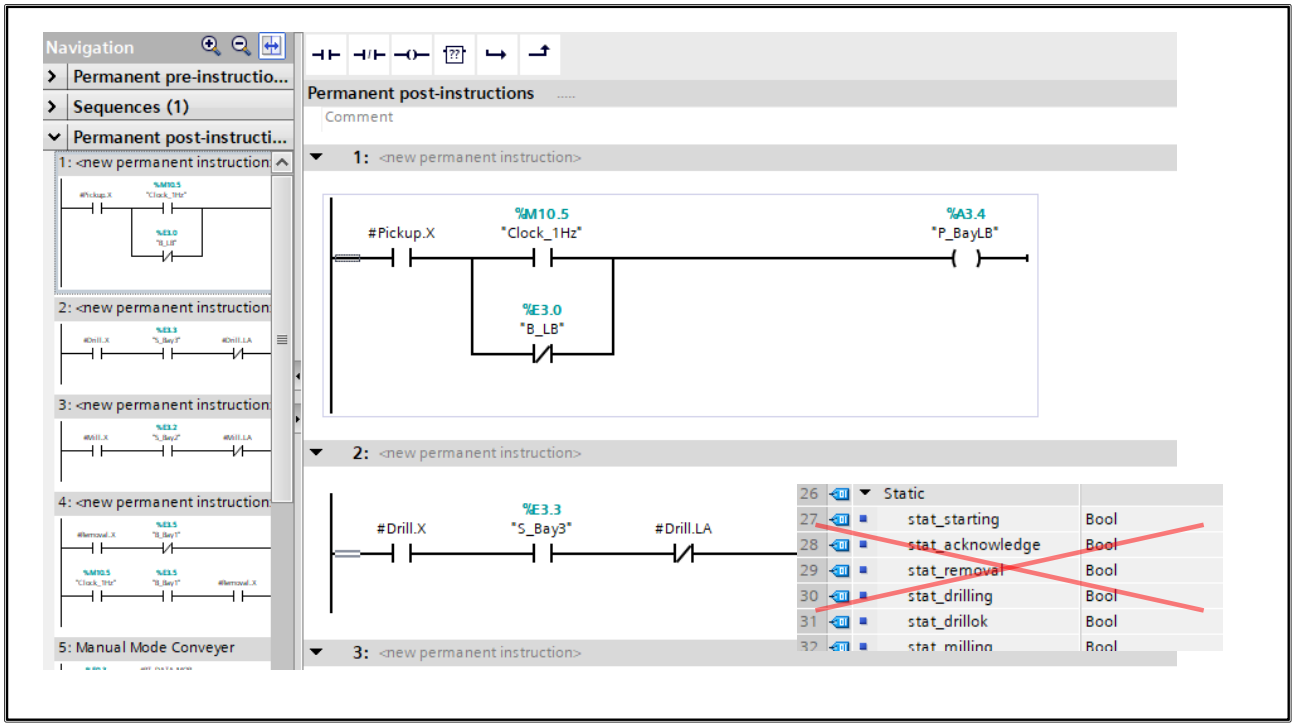
The GRAPH system generates a static signal **LA** (Lock Active) for the step with programmed interlock logic, which is active until this logic is **not true**. This signal can be transferred to the next transition as a negated query with designation **Si.LA**.

Note

If the option "Processing of all interlocks in manual mode" was selected in the "Block settings", this signal is also available to the interlock conditions of other steps and can be queried there. This also applies to program sections in the "permanent operations".



6.12. Exercise 1: Adapt FB Production



Task

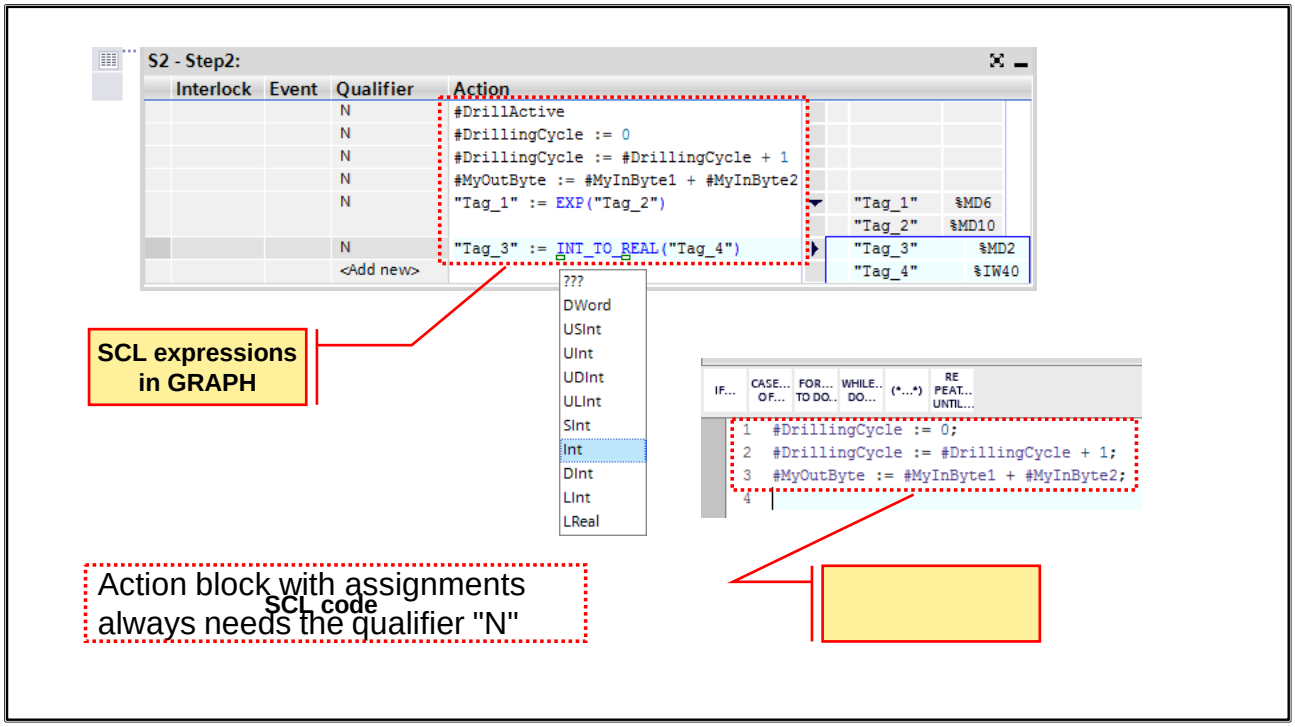
The step sequence (linear sequence) programmed in the last exercise will now be improved by adding a few things. Nothing to do with the sequence of actions is to be changed initially.

Some own static tags of your step sequence are to be replaced by internal static parameters.

Procedure

1. Delete your own tags from the block interface.
3. Start the improved program and test whether the previous functionality has been retained.

6.13. Use of SCL expressions in GRAPH



SCL expressions

Actions can be logically combined with simple arithmetic expressions. An action that contains an arithmetic expression always requires the N operation.

Additionally, the action can be made dependent on an event and/or logically combined with an interlock.

Assignments

Assignments with arithmetic operations can be subdivided into the following three groups:

- Direct assignments: They are entered according to the syntax "A := B".
- Assignments with integrated functions: They are entered according to the syntax "A:=func(B)". The integrated functions can, in turn, be subdivided into three groups:
  - Conversion functions such as: INT_TO_DINT, DINT_TO_REAL, ROUND, BCD_TO_NUM, etc.
  - Complex arithmetic functions such as SIN, COS, EXP, LN, ASIN, etc.
  - Other functions such as NOT, SWAP, RLDA, RRDA, etc.
- Assignments with arithmetic operations

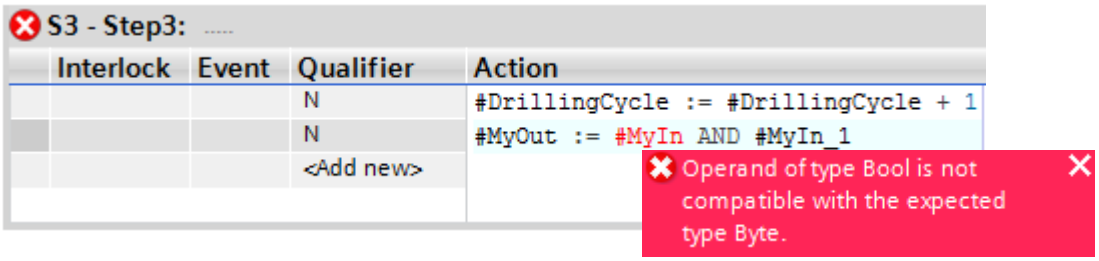
They are entered according to the syntax "A := B <operator> C".

-, *, / operators are the basic arithmetic operations +,

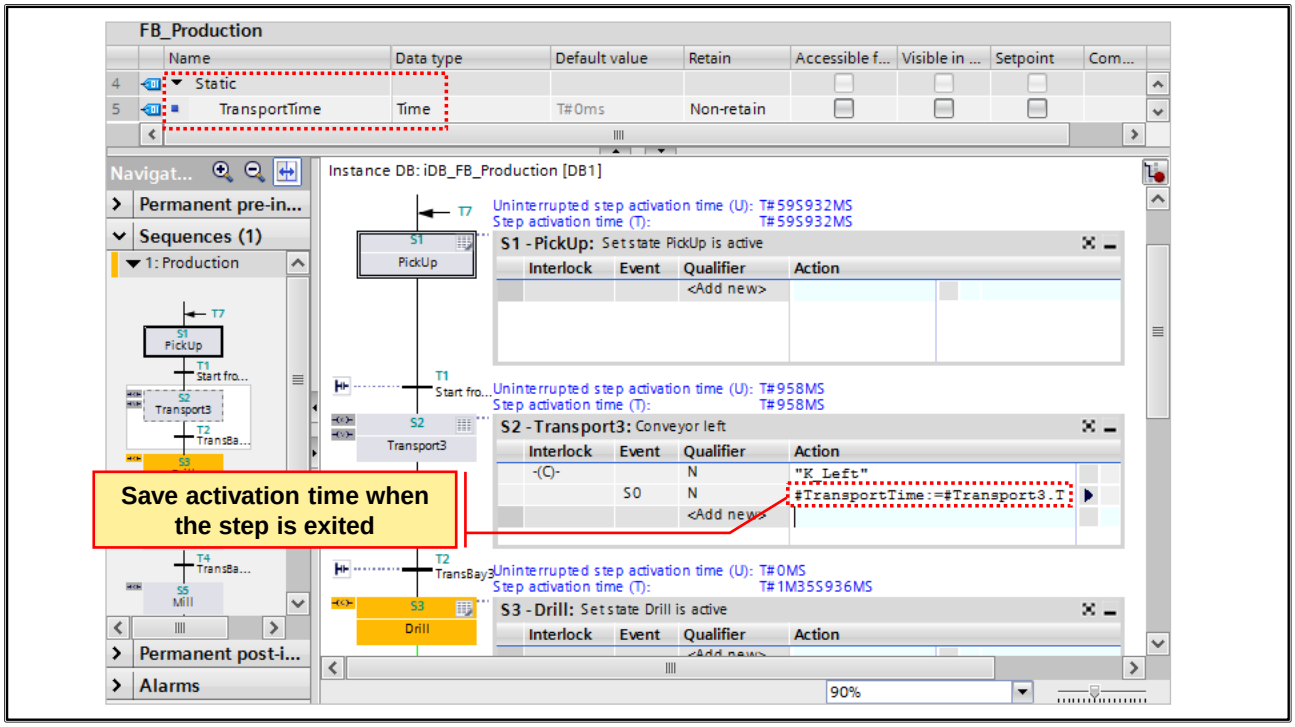
Note

For functions and operators, the data type of the assigned Operand A always defines the data type of the expression.

Boolean logic operations are currently not yet possible in GRAPH!



6.14. Running time of single steps



Running time

With GRAPH it is possible to store the running times of the single steps in any memory area. For each step that is located in the sequencer of the GRAPH function block, a step structure is created in the instance data block.

The structure element "StepX.**T**" contains the total step activation time. The element "StepX.**U**" contains the uninterrupted step activation time. With the help of arithmetic functions, this running time information can be stored in any memory area.

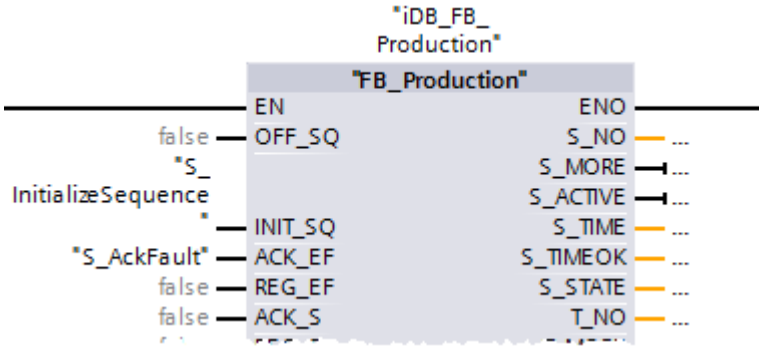
For this purpose, an action must be programmed for each step that copies the step activation time from the instance data block to the corresponding memory area on an event-dependent basis when the step is exited. For example:

S0 N "Memory area" := Step1.T

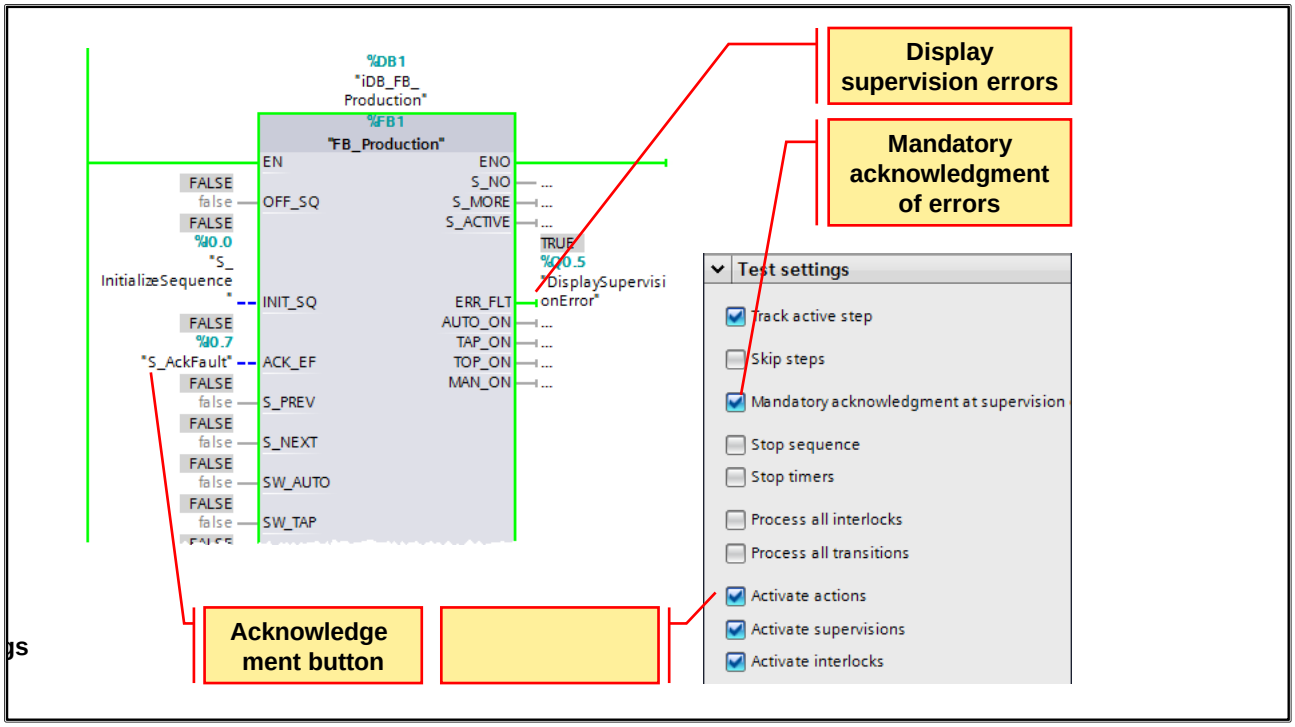
FB parameters

The option also exists to route the running times via the FB interface. The current running times of the currently active step are made available using the output parameters "S_TIME" and "S_TIMEOK" of the GRAPH function block.

In order to see these parameters when the Graph FB is called, this must be displayed with maximum interface parameters!



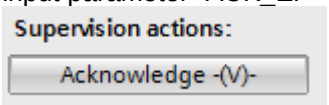
6.15. Mandatory acknowledgment of errors



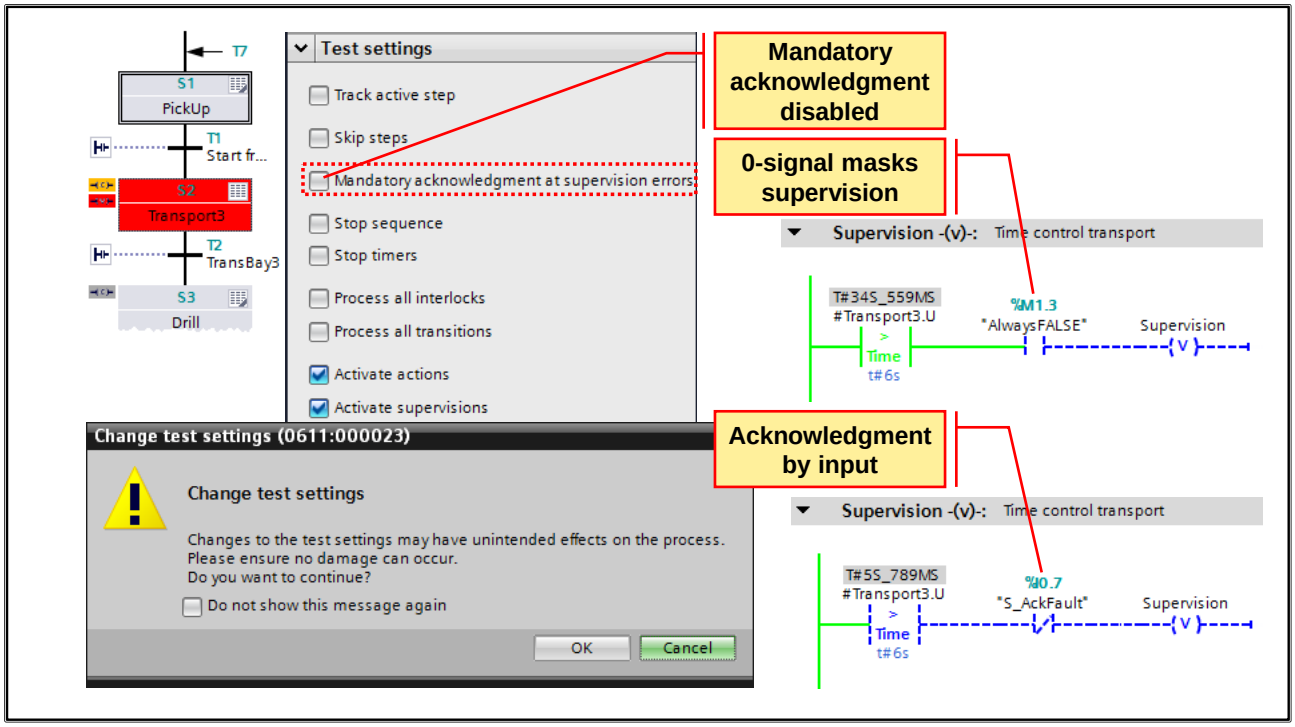
Requirements for mandatory acknowledgment of errors

The signaling and acknowledging of a supervision error requires the following settings in the "Test settings" dialog box:

- The "Mandatory acknowledgement at supervision errors" option is selected. The FB output parameter "ERR_FLT" is not influenced by this setting, however.
- This setting can be changed either via the application setting as default setting for new created blocks: (Options → settings → PLC programming → Graph) or via block properties → attributes, as specific setting for the current block.
- Supervision errors that occur during operation must then be acknowledged using either the input parameter "ACK_EF" or the sequence control "Acknowledge -(V)-".



6.16. Acknowledgment in the supervision logic

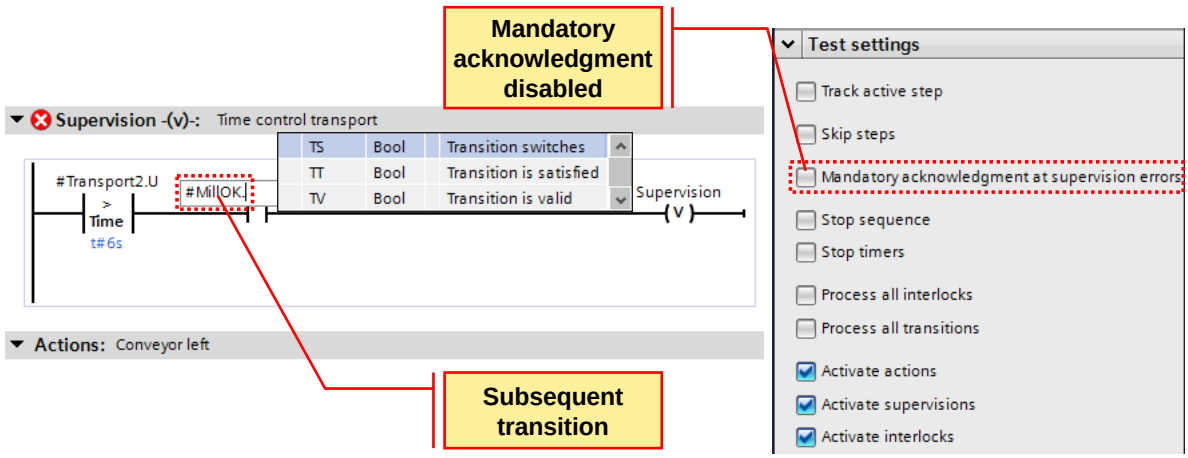


No mandatory acknowledgment

The following options for acknowledgment without explicit mandatory acknowledgment ("Mandatory acknowledgement at supervision errors" is disabled) via the step sequence block are available:

- Acknowledgment in the supervision itself (see slide)
- Self-acknowledgment when transition condition is true (see next slide)
- Other options for bypassing an acknowledgment, which are not directly related to the setting "Mandatory acknowledgement at supervision error" are:
- Step with fault is switched off and on again (S_OFF→ S_ON for the stopped step). This restarts the monitoring time.
- Reinitialize step sequence (INIT_SQ), i.e. restart the step sequence in the initial step.
- Switch off step sequence (OFF_SQ). The whole sequence is switched off and loses its memory, i.e. the sequence must be reinitialized or selectively set to a step.

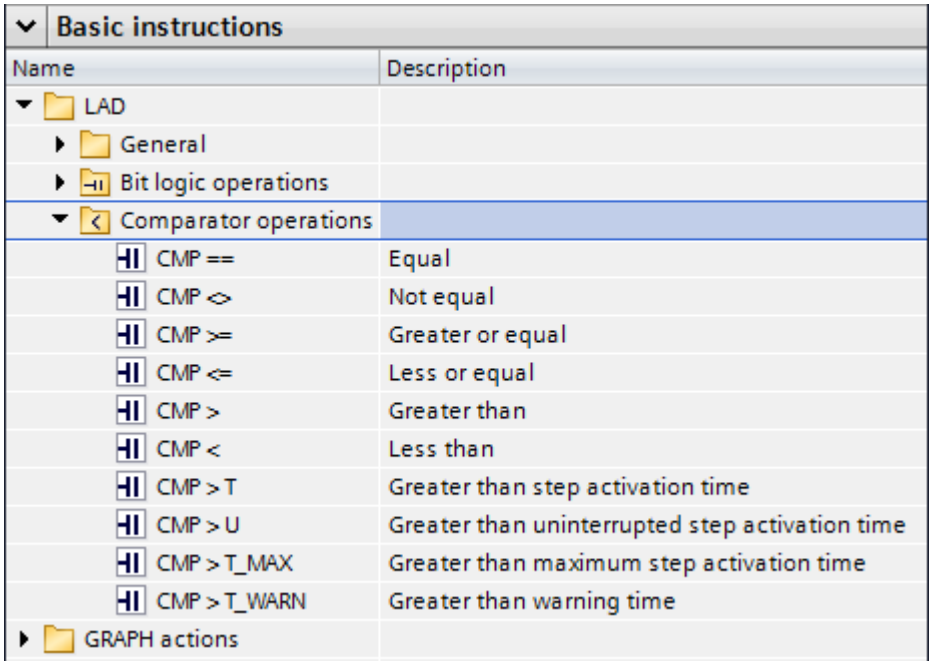
6.17. Query of transition states



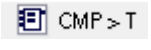
Step S4 (#Transport2) is time-monitored for 6 seconds in which case the Supervision can be self-acknowledged when the transition condition is true (TT=Transition True)
(Test setting: no mandatory acknowledgment).

Overview

The "Comparator" operation enables comparison of step running times and process values.



For programming of step monitoring times, ready-made elements (comparison boxes) are provided (available in Favorites) that facilitate the input:

- Insert time monitoring T 

- Insert time monitoring U

Example

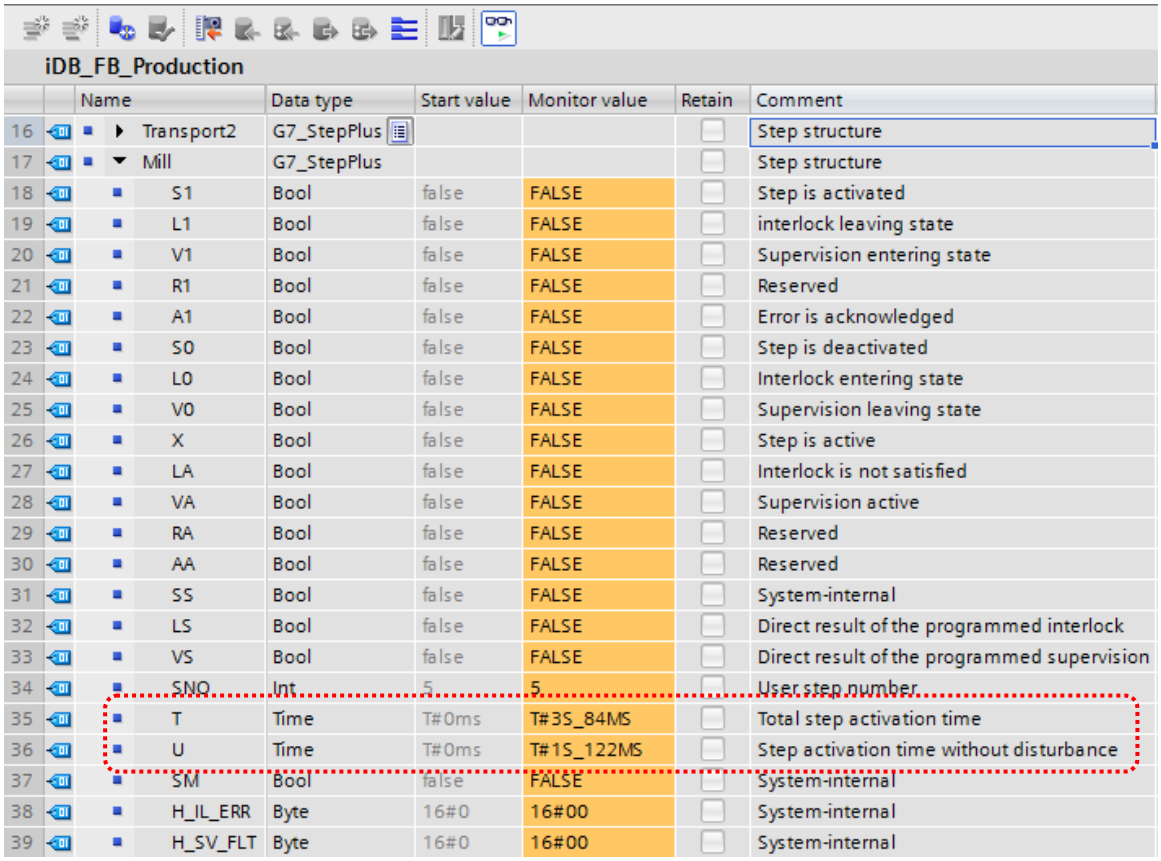
If the step activation time of step S4 "Transport 2" (#"Transport 2".U) exceeds the limit of 6 seconds, a fault is generated. When the associated transition condition is true, the comparison is interlocked by the negated query #Ini2.TT and the supervision condition is true again.

Thus, the sequence can be advanced via the acknowledgement or with self-acknowledgement. Which method has to be used is dependent on the mandatory acknowledgement setting.

Step times

For each active step, the time values are continually updated, saved in the instance DB and displayed above and to the left of the action block when the step sequence is monitored. The S5-compatible timers present in the S7-1500 are not used.

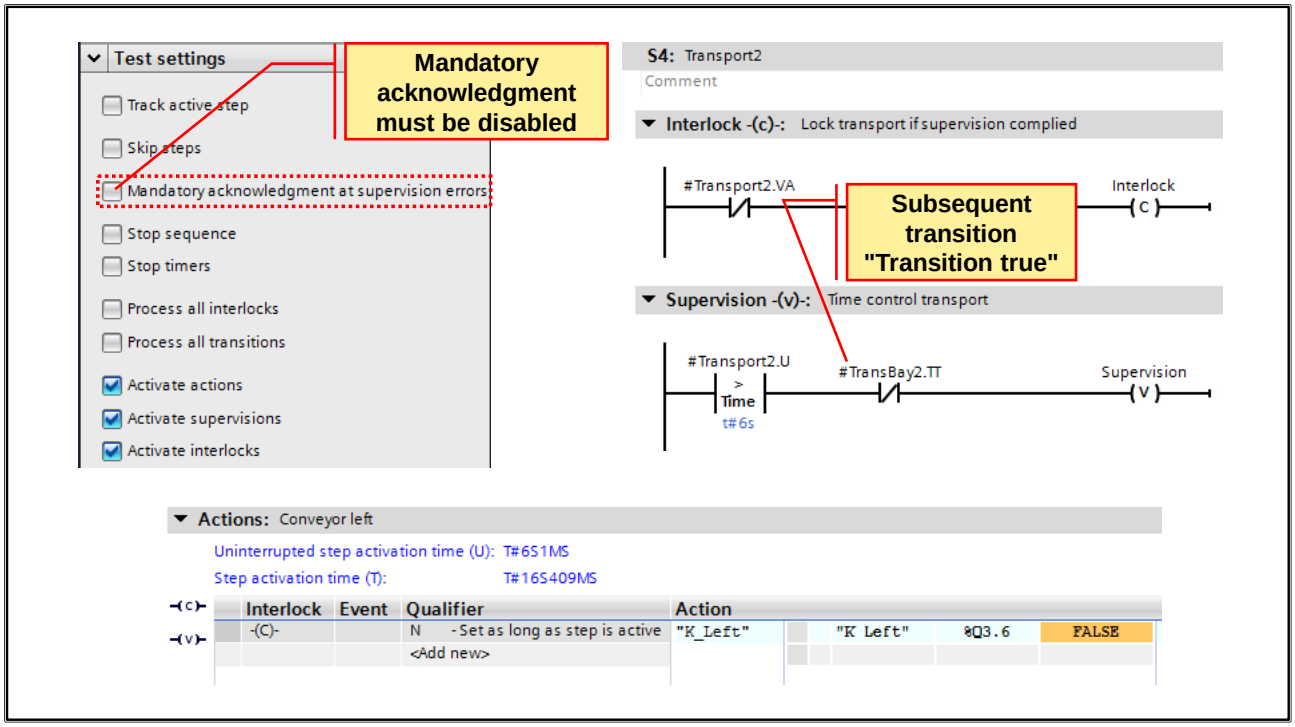
The times last needed are stored for each step in the instance data block until the next time the step is passed through. The times can be queried across steps.



iDB_FB_Production

	Name	Data type	Start value	Monitor value	Retain	Comment
16	Transport2	G7_StepPlus				Step structure
17	Mill	G7_StepPlus				Step structure
18	S1	Bool	false	FALSE		Step is activated
19	L1	Bool	false	FALSE		interlock leaving state
20	V1	Bool	false	FALSE		Supervision entering state
21	R1	Bool	false	FALSE		Reserved
22	A1	Bool	false	FALSE		Error is acknowledged
23	S0	Bool	false	FALSE		Step is deactivated
24	L0	Bool	false	FALSE		Interlock entering state
25	V0	Bool	false	FALSE		Supervision leaving state
26	X	Bool	false	FALSE		Step is active
27	LA	Bool	false	FALSE		Interlock is not satisfied
28	VA	Bool	false	FALSE		Supervision active
29	RA	Bool	false	FALSE		Reserved
30	AA	Bool	false	FALSE		Reserved
31	SS	Bool	false	FALSE		System-internal
32	LS	Bool	false	FALSE		Direct result of the programmed interlock
33	VS	Bool	false	FALSE		Direct result of the programmed supervision
34	SNO	Int	5	5		User step number
35	T	Time	T#0ms	T#3S_84MS		Total step activation time
36	U	Time	T#0ms	T#1S_122MS		Step activation time without disturbance
37	SM	Bool	false	FALSE		System-internal
38	H_IL_ERR	Byte	16#0	16#00		System-internal
39	H_SV_FLT	Byte	16#0	16#00		System-internal

6.18. Self-acknowledgment when transition condition is true



Fault handling with self-acknowledgement

The example shown was programmed in such a way that an interlock fault occurs simultaneously when a supervision error (time monitoring T#6s) occurs. As a result of this, the conditioned assignment with RLO=0 is executed and thus the output "K_Left" (%Q3.6) is disabled.

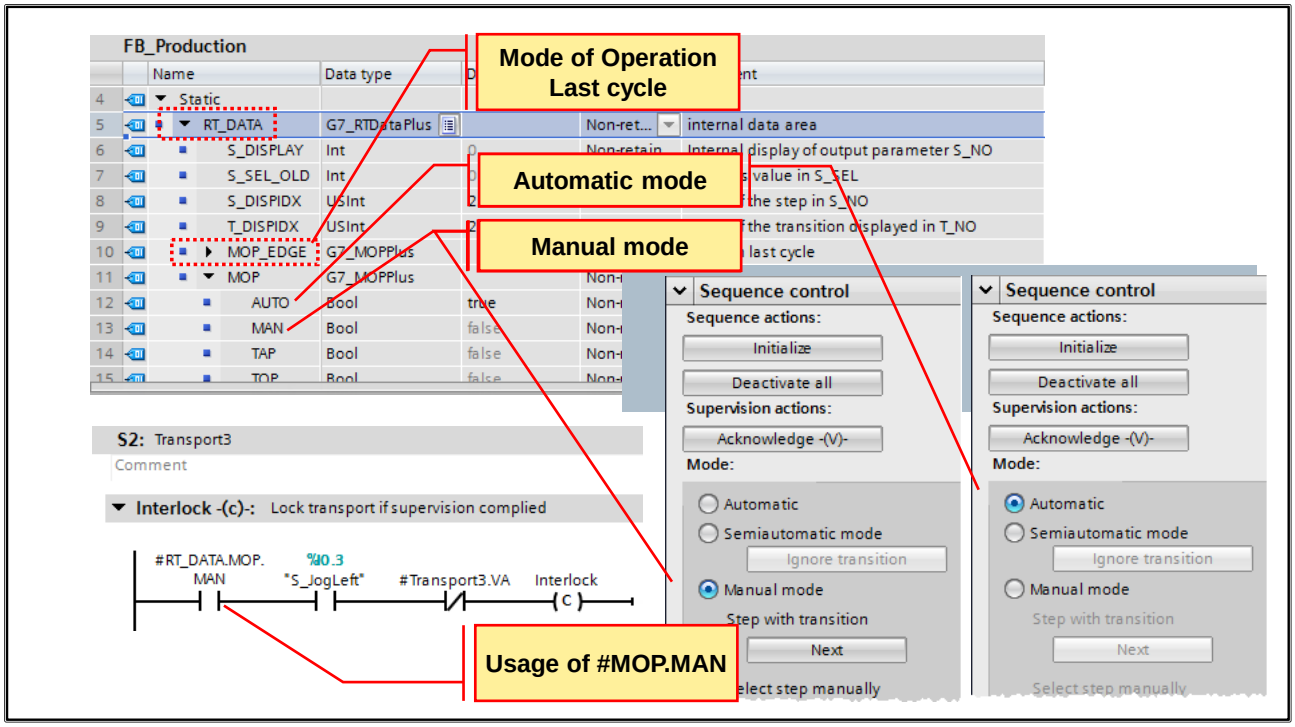
#Transport2.VA contains the supervision error display for step S4. The output "K_Left" (%Q3.6) is only set as long as S4 is active and **no** supervision error is present.

If the subsequent transition condition is true, the supervision error is acknowledged automatically by a query of the signal "#TransBay2.TT" (Mandatory acknowledgment at supervision errors must be disabled).

Note

If an automatic acknowledgment is desired, this must be integrated in the supervision (as described above under "Self-acknowledgement when transition condition is true") for each step or the supervision must not be used.

6.19. MOP – Mode of Operation



Operating modes

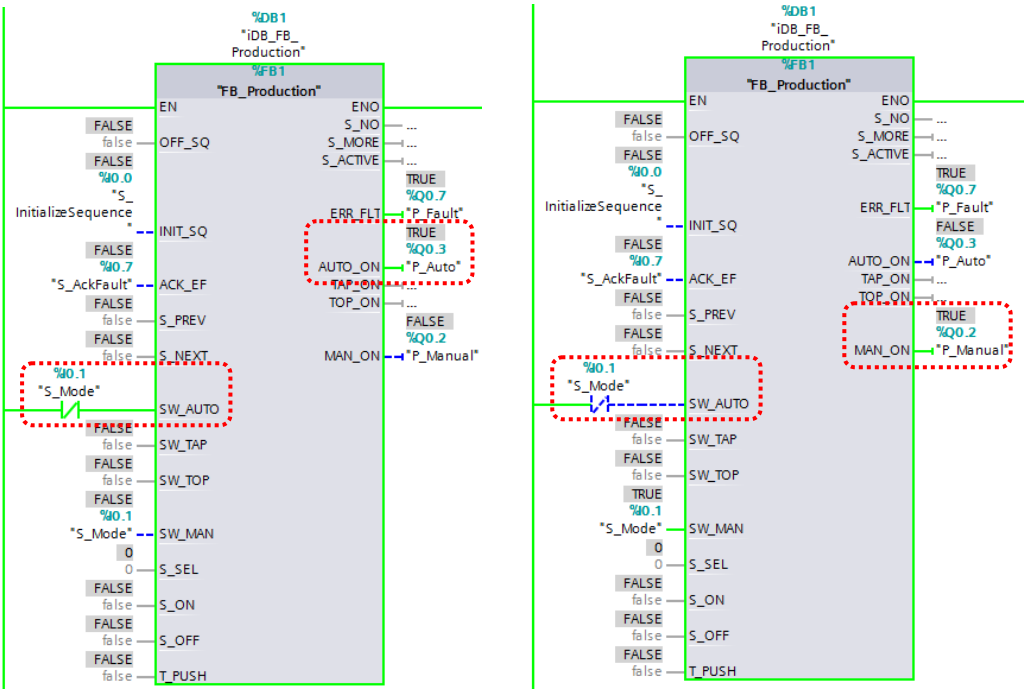
The static parameter area of the sequence block has a structure named MOP (Mode of Operation) that contains a large amount of information about states, requests, etc. from the sequence.

The slide shows the integration of the feedback MAN #MOP.MAN in the interlock of step 2 (Transport3). As shown in the figure, Manual mode can be selected using the sequence control in online test mode.

Alternatively, the operating modes can also be controlled externally using the FB parameters. This requires a corresponding interconnection of the sequence block in OB1.

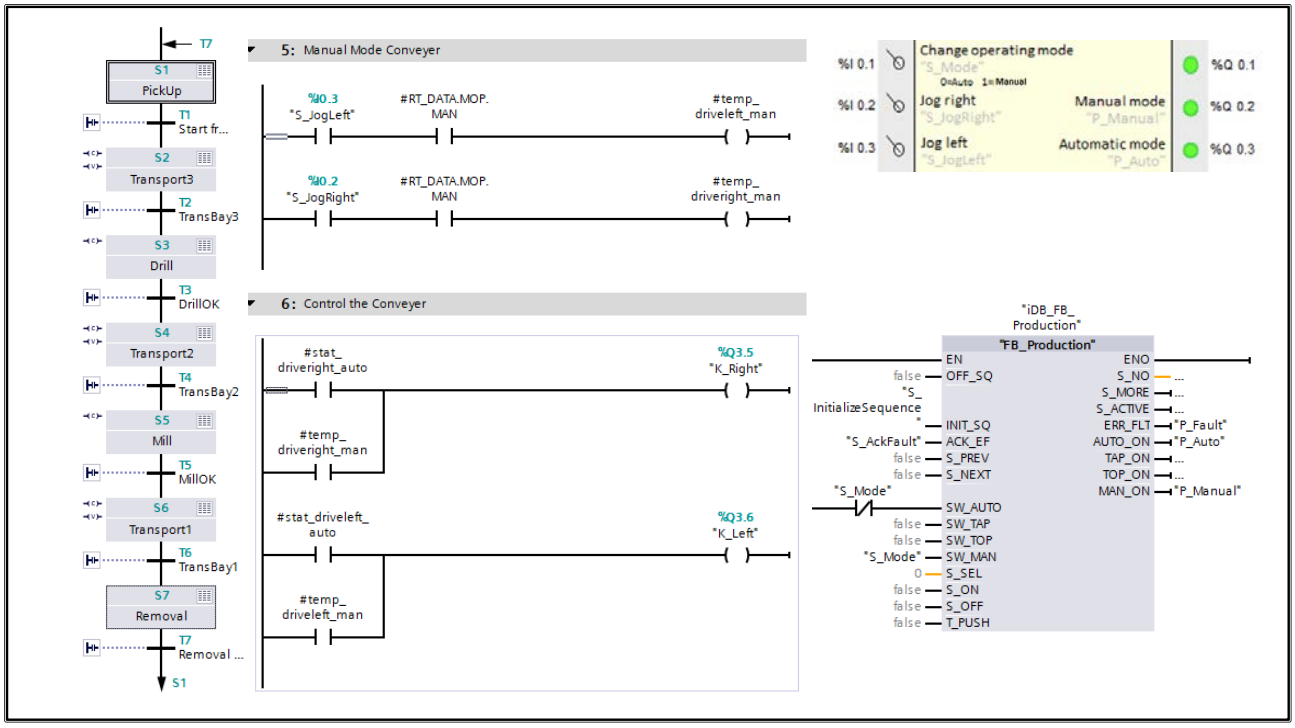
Example

The example below shows the control of operating modes by the passing of FB parameters when the GRAPH FB is called in OB1:



The figure on the left shows the selection and feedback of automatic mode. The figure on the right shows the selection and feedback of manual mode.

6.20. Exercise 2: Integrate automatic and manual modes



Task

The step sequence (linear sequence) programmed in the last exercise will now be expanded to include a mode switch.

The GRAPH FB is to be switched to automatic mode or manual mode using an external interconnection "Switch mode" (%I0.1 – S_Mode):

- The sequence is to function as usual in automatic mode.
- In manual mode, the conveyor belt can be moved using the "Jog right" (%I0.2 – JogRight) and "Jog left" (%I0.3 - S_JogLeft) pushbuttons. This is also used to remedy a possible blocking of the conveyor or to move a part to a desired position.

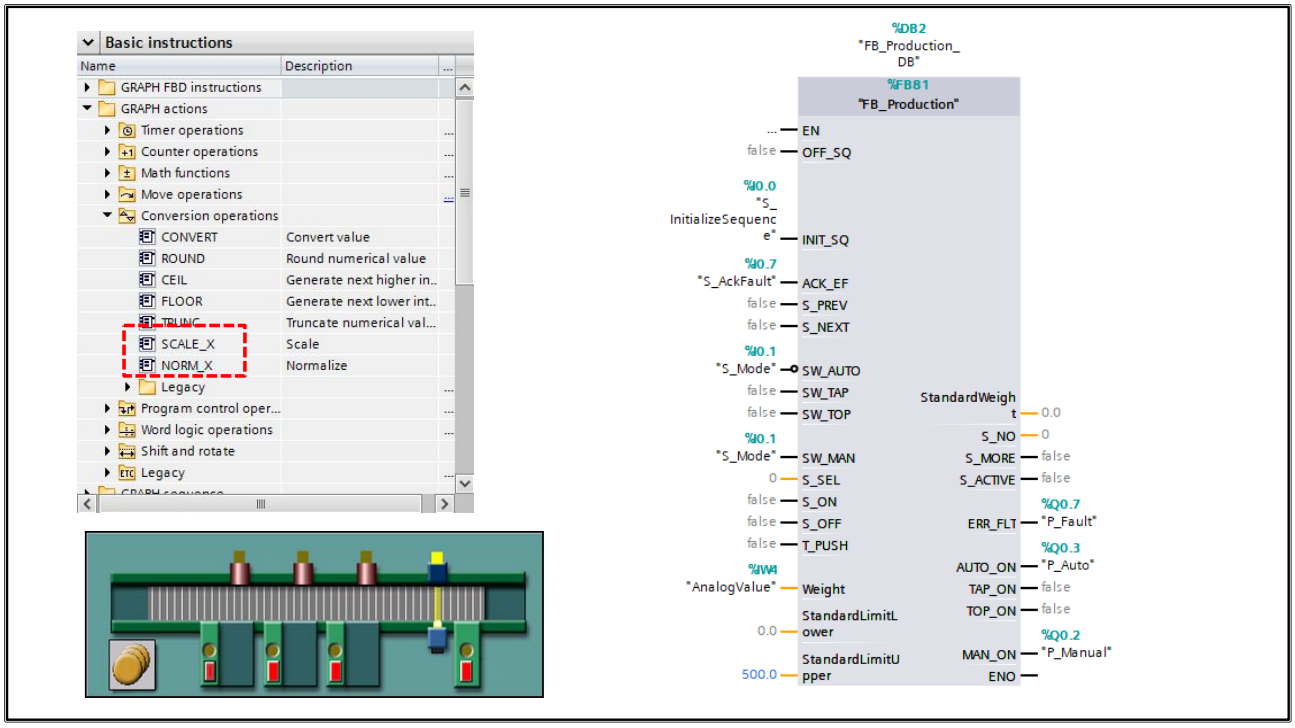
Both operating modes are signaled back using the respective LEDs "Manual mode" (%Q0.2 – P_Manual) and "Automatic mode" (%Q0.3 – P_Auto).

All newly created tags are to be created in the "Simulator" PLC tags.

Procedure

1. Change your program at the appropriate places so that the above-described function for forward (right) and reverse (left) running of the conveyor in manual mode is achieved. Use the inputs "%I0.1", "%I0.2" and "%I0.3" from your operator panel for this.
   Automatic mode of the conveyor for transporting the blank towards the left is to continue functioning as usual.
2. Use the internal tags available to you in the static parameter area "MOP" for the recognition of the current operating mode.
   If you need auxiliary tags, create these as needed in either the static or temporary area of the interface.
3. Interconnect the feedback of the operating modes of the step sequence to the outputs "%Q0.2" and "%Q0.3" of the operator panel.
4. Test whether the step sequence runs as usual in automatic mode and then test the manual operation with the jog function in each step.

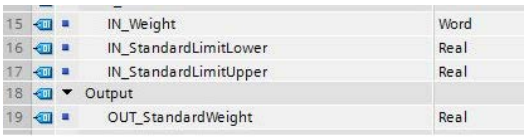
6.21. Exercise 3: Event-driven weight acquisition



Task

- Before the machining of the blank begins, it must be determining whether the weight of the blank is within the tolerance range of the production process. The blank is therefore weighed in the "Pick-up" step. The weight can be between 0 and 500 kg.
- If the weight of the workpiece is outside the specified tolerance limits, 100 to 400 kg, the production operation is not started. In order for this to be signaled to the user, the LED "P_BayLB" (%Q3.4) turns off in this case.
- If the weight of the blank is within the prescribed tolerance of 100 to 400 kg, the step sequence can be started as usual.
- The weight value is simulated by an analog module integrated in the training device (%IW4 – Analog value). By setting the potentiometer between 0…10 V, a weight value between 0 and 500 kg is simulated.
- (Alternative – instead of analogue value calculation) Use a MDxxx to control the real value for the current weight directly from a watch table or from an I/O field at the HMI.

1. Transfer the Norm_X and Scale_X functions from the instruction section to the corresponding action block and consult the online help for information about the parameter assignment.



2. Do not call the functions as an event-driven function. That is, the workpiece will be re-measured in each cycle in the "Pick-up" step.

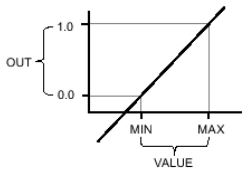
3. The input parameters and the output parameter are to appear as user-defined parameters in the GRAPH FB and be initialized and removed there. Also create the input tag %IW4 – Analog value in the "Simulator" tag table.

NORM_X: Normalize

Description

You can use the instruction "Normalize" to normalize the value of the tag at the VALUE input by mapping it to a linear scale. You can use the parameters MIN and MAX to define the limits of a value range that is applied to the scale. Depending on the location of the normalized value in this value range, the result at output OUT is calculated and stored as a floating-point number. If the value to be normalized is equal to the value at input MIN, output OUT returns the value "0.0". If the value to be normalized is equal to the value at input MAX, output OUT returns the value "1.0".

The following figure shows an example of how values can be normalized:



The "Normalize" instruction works with the following equation:

OUT = (VALUE - MIN) / (MAX - MIN)

The enable output ENO has the signal state "0" if one of the following conditions applies:

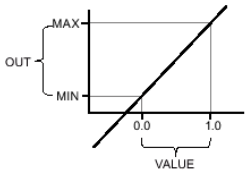
- Enable input EN has the signal state "0".
- The value at input MIN is greater than or equal to the value at input MAX.
- The value of a specified floating-point number is outside the range of the normalized numbers according to IEEE-754.
- The value at input VALUE is NaN (result of an invalid arithmetic operation).

SCALE_X: Scale

Description

You can use the "Scale" instruction to scale the value at the VALUE input by mapping it to a specified value range. When the instruction "Scale" is executed, the floating-point value at input VALUE is scaled to the value range, which is defined by the parameters MIN and MAX. The result of the scaling is an integer, which is stored at output OUT.

The following figure shows an example of how values can be scaled:



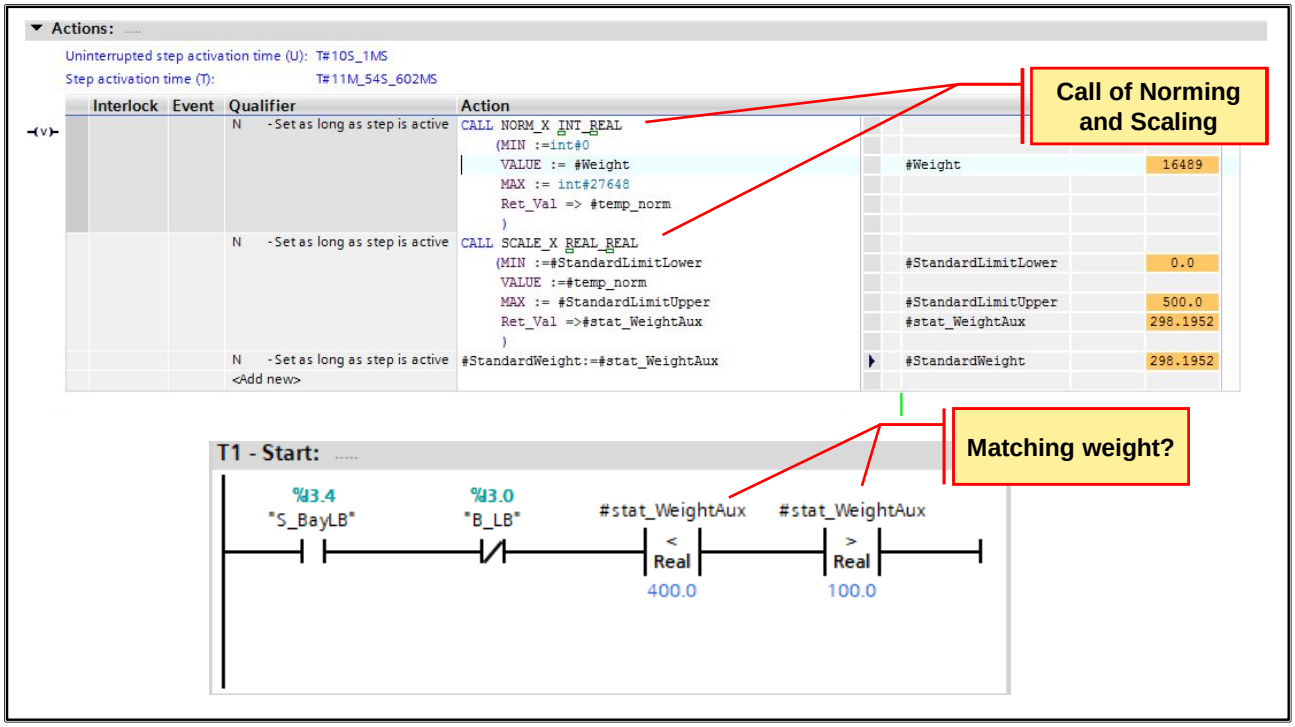
The "Scale" instruction works with the following equation:

OUT = [VALUE * (MAX - MIN)] + MIN

The enable output ENO has the signal state "0" if one of the following conditions applies:

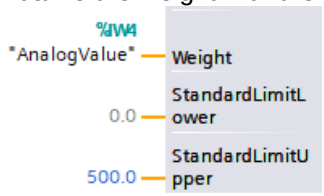
- Enable input EN has the signal state "0".
- The value at input MIN is greater than or equal to the value at input MAX.
- The value of a specified floating-point number is outside the range of the normalized numbers according to IEEE-754.
- An overflow occurs.
- The value at input VALUE is NaN (Not a number = result of an invalid arithmetic operation).

6.22. Exercise 3: Continuation

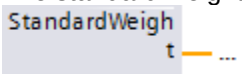


Procedure

4. Program the necessary interlocks in the permanent post-instructions and in the transition after the "Pick-up" step. If necessary, create temporary or static tags in the block interface.
5. After the program expansion, the sequence must be recompiled. Also check the FB call in OB1 for time stamp conflicts and update these.
6. Set the initialization of the high and low standard limits to 0.0 and 500.0 by using constants. Initialize the weight with the analog value from the potentiometer (%IW20 - Analog value).

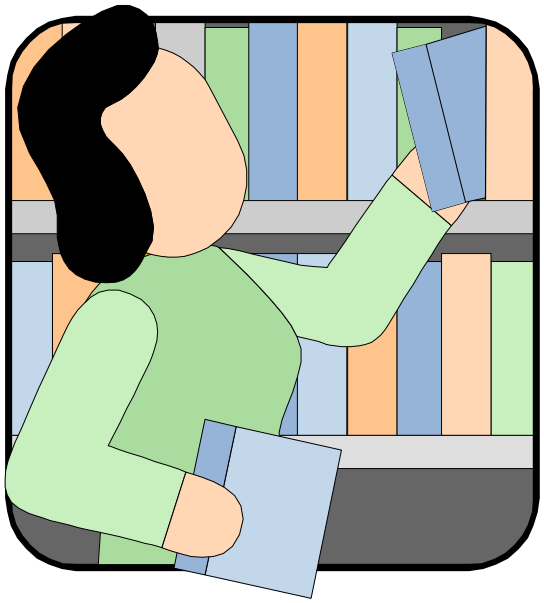


7 The standard weight will be removed in the next exercise. Do not interconnect this.



8. Compile your project.
9. Download and test the program under various conditions.

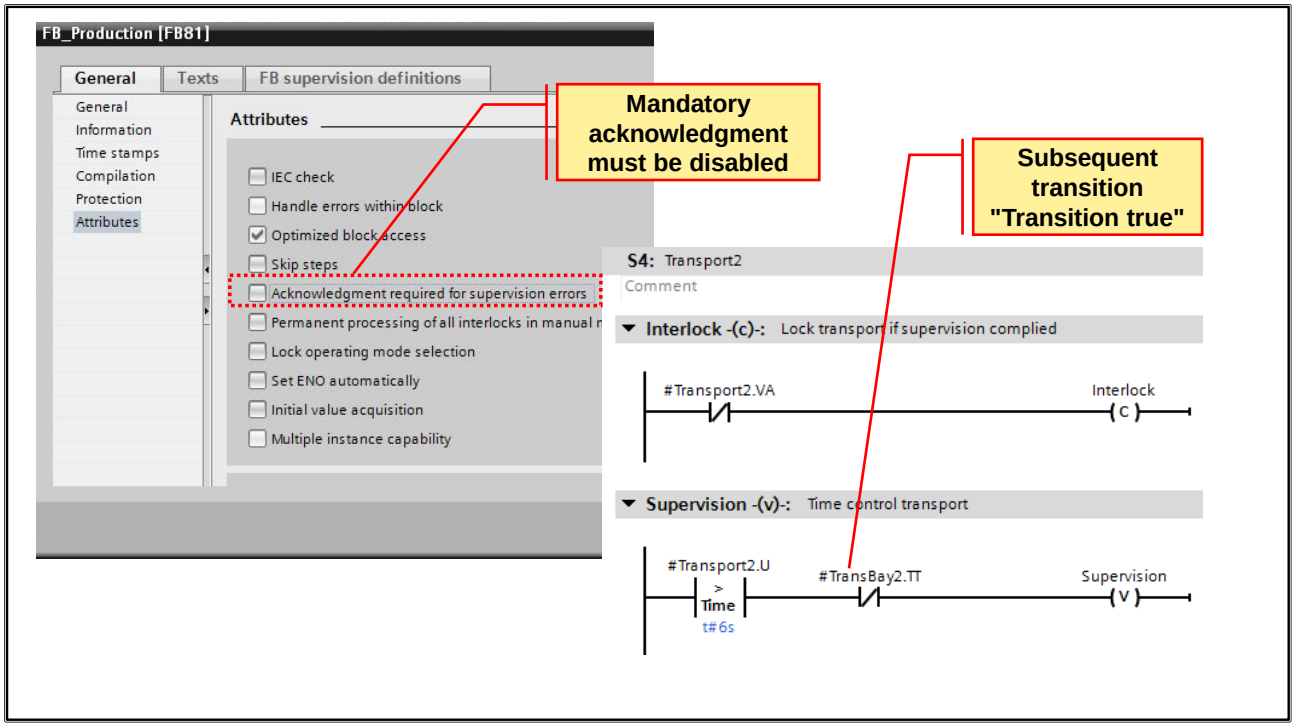
6.24. If you want to know more



Note

The following pages provide additional information or serve as a reference that can be consulted for more complete information on a subject.

6.25. Exercise 4: Self-acknowledgment of supervision errors



Task

Supplement your program to include a self-acknowledgment of supervision errors.

Procedure

1. Make the necessary program changes in the appropriate steps.
2. Adapt your test settings for the self-acknowledgment.
3. Download your changes to the PLC and test them!

Table of contents

7

7. Branches and Jumps 7-2

7.1. Overview 7-3
7.2. Alternative branches and priority 7-4
7.3. Simultaneous branches 7-5
7.4. Application of a simultaneous branch 7-6
7.5. Exercise 1: Use of alternative branch for reverse transport following invalid final weight 7-7
7.6. Exercise 2: Addition of simultaneous branch 7-8
7.7. Exercise 3: Addition of 2nd sequence 7-9
7.8. If you want to know more 7-10
7.9. Exercise 4: Count bad parts 7-11

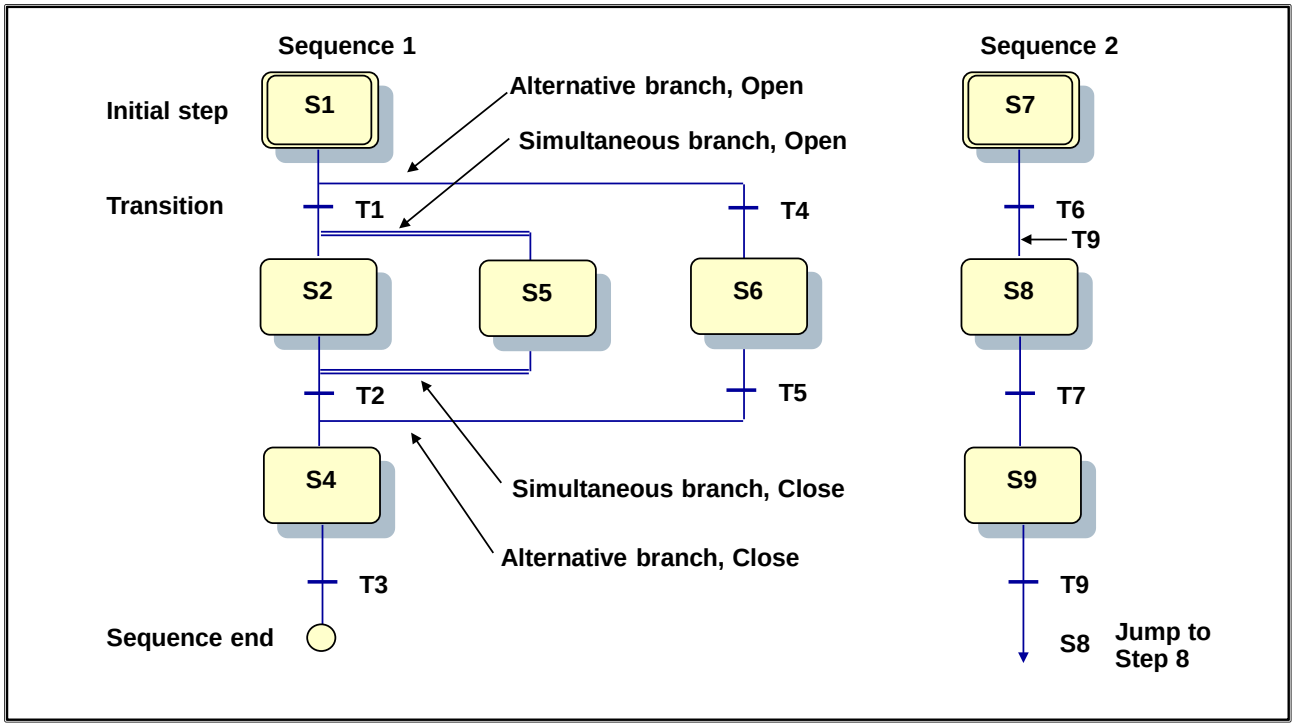
7. Branches and Jumps

The participant should ...

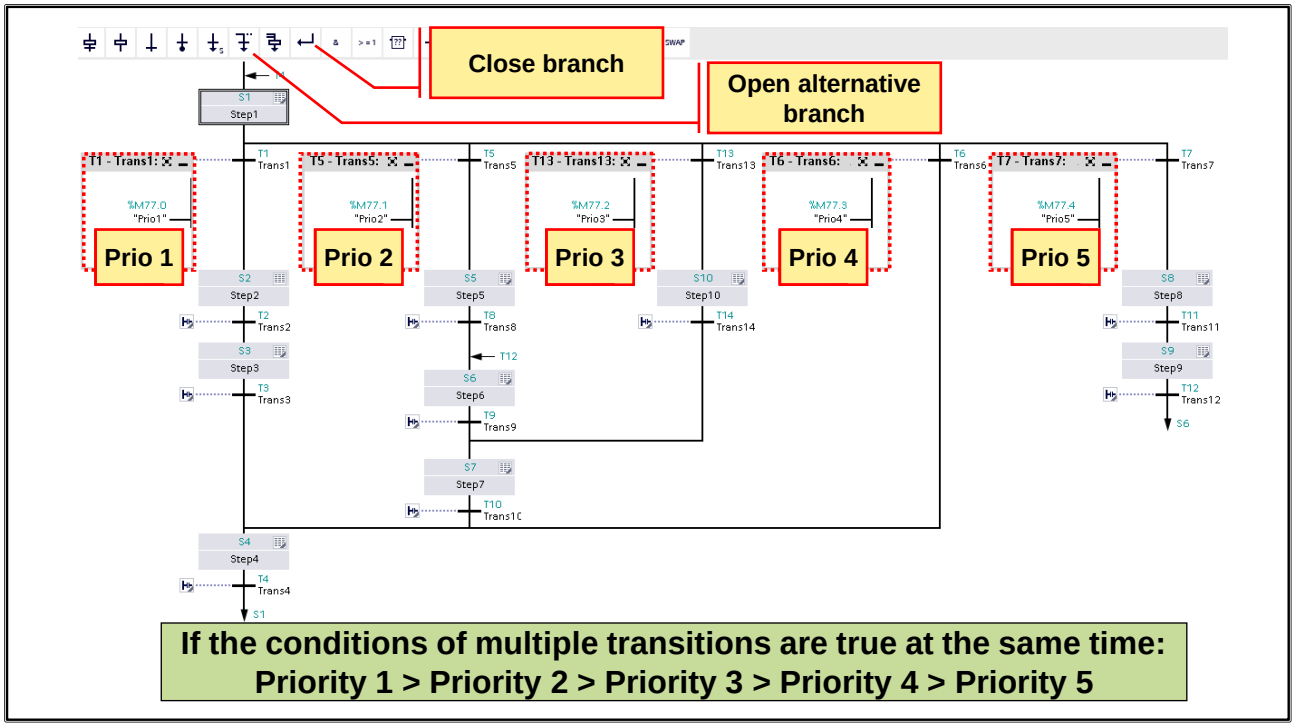
… Gain knowledge about alternative branches and simultaneous branches and be able to use them

... become acquainted with sequence complexes (sub-sequences)

7.1. Overview


Sequence 1
Sequence 2
Initial step
S1
Alternative branch, Open
Simultaneous branch, Open
S7
Transition
T1
T4
T6
T9
S2
S5
S6
S8
T2
T5
T7
S4
Simultaneous branch, Close
Alternative branch, Close
S9
T3
T9
Sequence end
S8
Jump to
Step 8

7.2. Alternative branches and priority


Close branch
Open alternative branch
Prio 1
Prio 2
Prio 3
Prio 4
Prio 5
If the conditions of multiple transitions are true at the same time:
Priority 1 > Priority 2 > Priority 3 > Priority 4 > Priority 5

Alternative branches

This element is used for alternating branching of the step sequence based on various conditions. An alternative branch always starts between a step and a transition. As shown on the slide, multiple alternative branches can branch from the same place.
An alternative branch can contain any number of steps and transitions. It can then lead back to the original branch or to another branch. It can also conclude with a jump or a sequence end.
The left branch is also called the main branch and has the highest priority.

A possible application is, for example, in manufacturing processes in which the further behavior of the plant (of the program) depends on the production quality of a produced part. A decision must be made between:

- Workpiece OK → continue with complete machining (left branch) or
- Workpiece not OK → do not machine, move to scrap line (right branch)

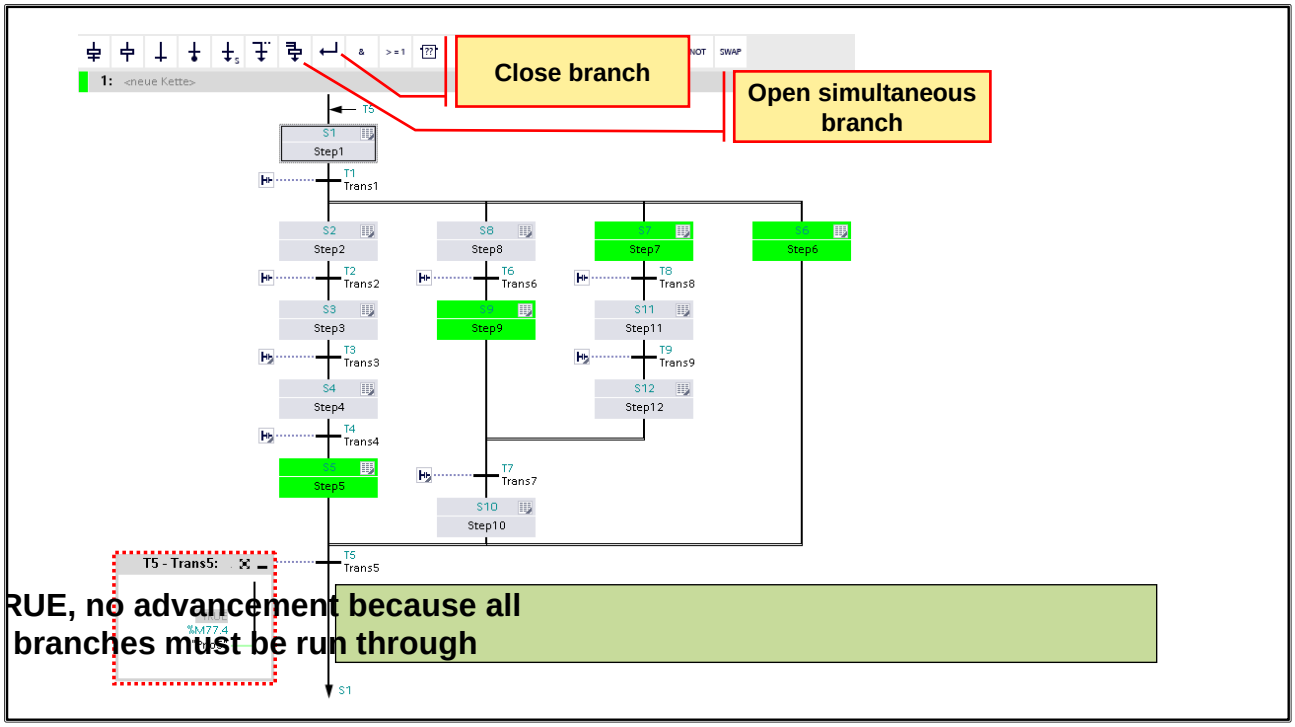
Only one alternative branch is activated at a time in the step sequence.

Limit case

If conditions of multiple opening transitions are true simultaneously, the branch located on the far left (main branch) always has highest priority. The remaining branches are prioritized in decreasing order from left to right

.

7.3. Simultaneous branches



Simultaneous branches

The arrangement of a simultaneous branch enables synchronous processing of multiple steps in parallel. A simultaneous branch always starts between a transition and a step. When the transition condition is true, all subsequent parallel steps are activated.

Starting from this time, the individual branches run fully independently. The set operating mode (automatic, semiautomatic or manual mode) affects all branches alike. A simultaneous branch must end with the closing into another branch. A sequence end or a jump cannot conclude a simultaneous branch.

The subsequent transition does not become valid until all branches have activated their last steps before converging. In so doing, the last steps of the faster branches in each case stay active until the last branch has reached its step before the closing branch.

Only when all steps before the convergence of the simultaneous branches are active will the subsequent transition (after the closing branch) become valid and, when its condition is true, be able to - according to graphical flow - advance the sequence in the left branch.

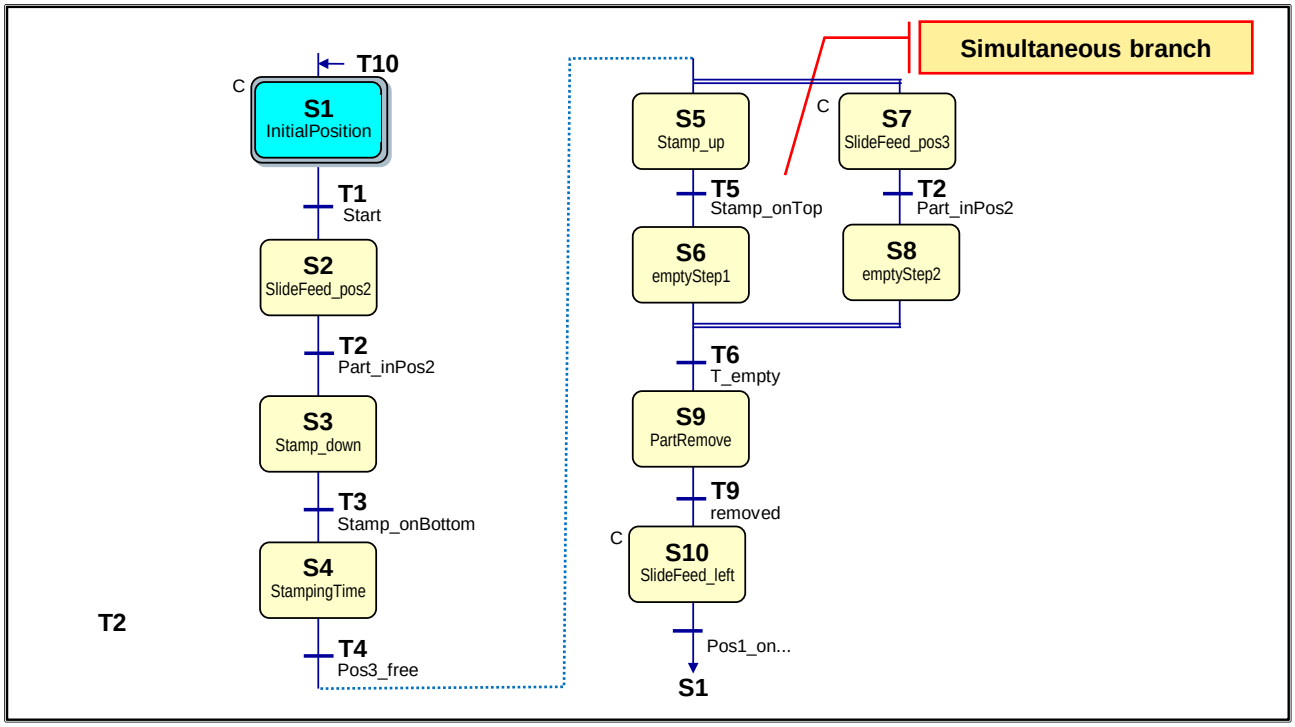
Note

A simultaneous branch corresponds to a synchronized AND branch. The start and end times of the branches are synchronized by the system (no additional S7 cycle).

As a result, all simultaneous branches are started simultaneously and upon closing all branches must again be closed simultaneously at the subsequent left branch!

If necessary, a "Wait step" can be used in branches to enable error-free resynchronization.

7.4. Application of a simultaneous branch



Implementation

The above figure shows a simplified overview representation of a "Stamping machine" program.

After actuation of a start signal (requires press to be in initial position), the slide feed is moved to position 2 in step S2 and delivers a part to the location below the press.

This activates step 3 and the press moves to the bottom stamp position. In step S4, the actual stamping operation takes place, a visual or acoustic warning could be output here as an action.

After the stamping time expires and when the removal position is unoccupied, the slide feed can transport the pressed part to position S3 and, simultaneously, the stamp can be moved to the top again.

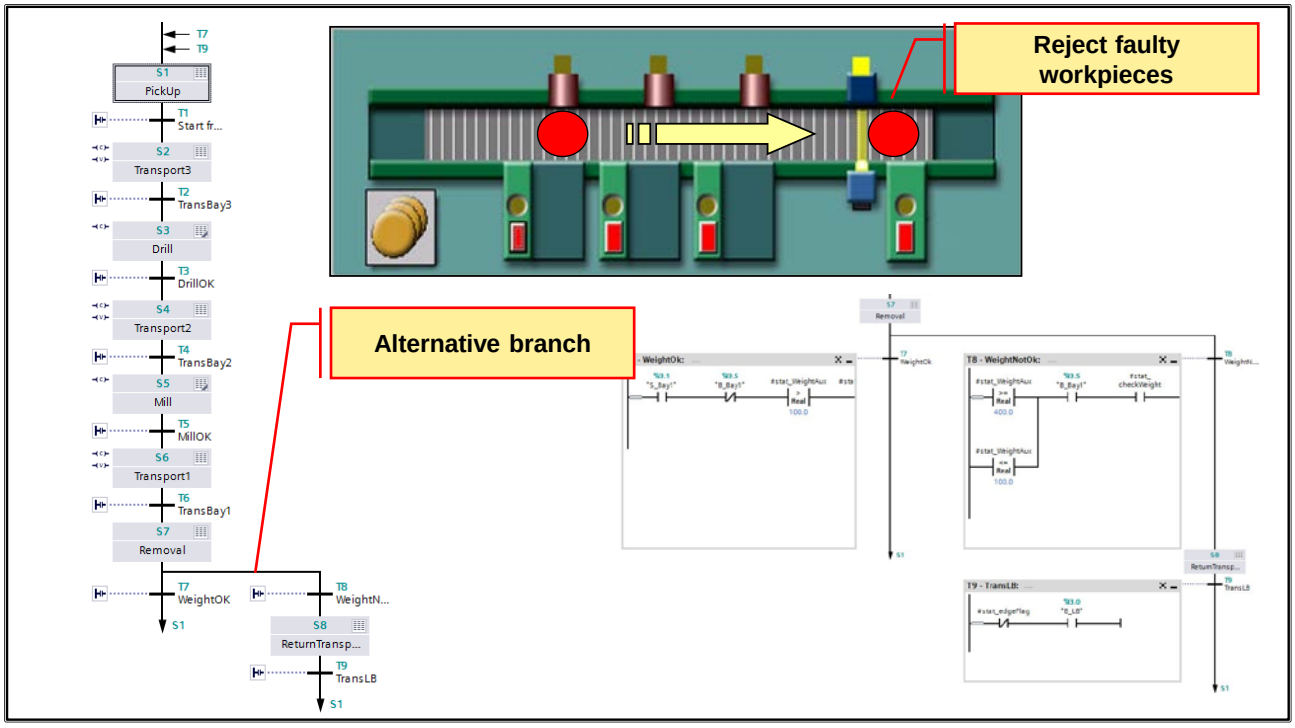
After the stamping operation, it is clear that the two subsequent processes "Stamp up" and "Slide feed forward" can run in parallel with regard to cycle time optimization.

In the application, once the bottom stamp position is exited, the slide feed moves forward (after a slight delay) So that each movement can be monitored separately, separate transitions are necessary.

But since only one shared transition is valid after the branches converge, a so-called wait step (empty step) must be inserted at the end of each branch.

The wait step is used to disable the preceding steps and to prepare the branches for a synchronized closing without a fault message.

7.5. Exercise 1: Use of alternative branch for reverse transport following invalid final weight



Task

The production process is to be changed as follows:

1. Because the weight of the blank can be reduced during machining as a result of drilling and milling, a second weight measurement will be implemented in the "Removal" step. This will be executed **once** when the step is activated.
2. If the weight measurement yields a weight value outside the prescribed tolerance limits (**100 to 400 kg**), the blank is to be transported in the reverse direction (to the right) through the light barrier.
3. If the weight measurement yields a weight value within the tolerance limits, the step sequence is to proceed as usual and the user can remove the blank. After confirmation using the pushbutton at bay 1, a new production operation starts.

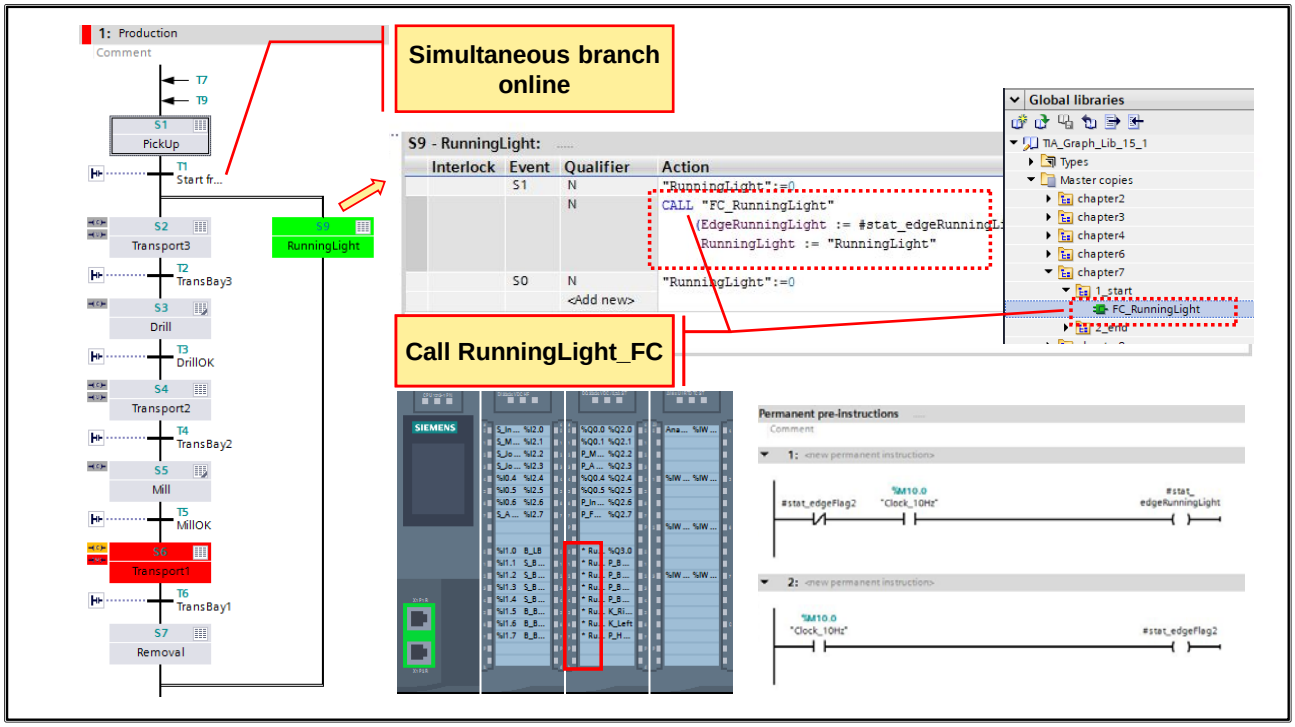
After the operation to sort out bad parts is complete, the sequence is to be brought back to the initial position.

Procedure

1. Insert an alternative branch at the appropriate place in your step sequence and make the necessary additions of steps and transitions.
2. Add the necessary parameters to the block interface of the GRAPH FB.
3. Organize the jump back to the initial position when the operation to sort out bad parts is finished.
4. The blank is to pass through the light barrier during the reverse transport. For this, program an edge evaluation at the appropriate place in the GRAPH FB.
5. The weight measurement in the "Removal" step takes 3 seconds. Incorporate the corresponding condition and the subsequent transitions.

Start the program again after the update in OB1 and test it.

7.6. Exercise 2: Addition of simultaneous branch



General

Because the conveyor model has only one linear motor with one axis, a practical example of a simultaneous branch on the conveyor model is not possible.
For this reason, a running light on the output module of the PLC will be started in parallel with the processing of the step sequence in this example.

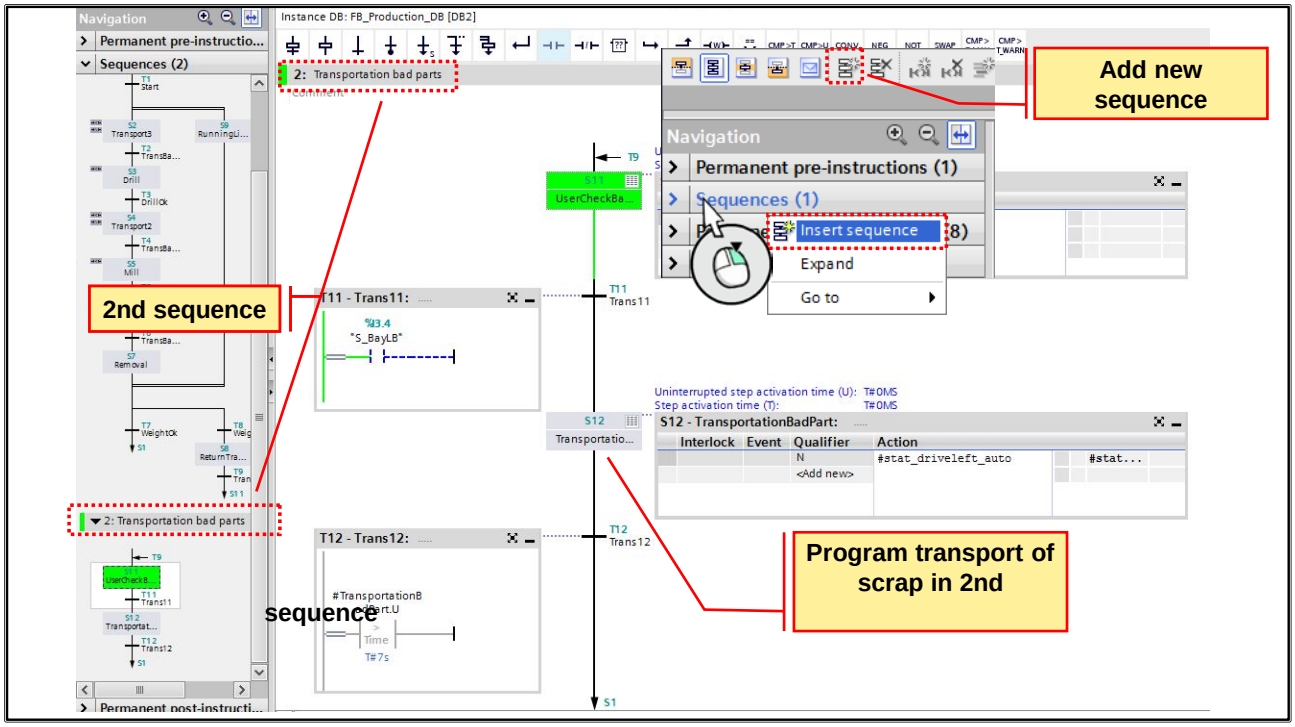
Task

After the start of the production process and once the "PickUp" step is exited, a running light is to indicate the active machining operation in parallel with the machining of the blank. This running light is output on the output module DQ 32x24VDC/0.5A ST_1 on the second byte (%QB1). The running light always starts with the first bit and is extinguished when the production process is finished, i.e. the "Removal" step is exited.
The running light is to be toggled at 10 Hz.

Procedure

1. Expand the sequence to include a simultaneous branch with the "Running light" step.
2. Open and close the simultaneous branch as shown on the slide.
3. Drag the "FC_RunningLight" block directly from the also automatically added to your "Program blocks" folder. Afterwards, interconnect the "RunningLight" byte (%QB1), which you created previously in the "Simulator" PLC tag table, with the InOut parameter of the FC.
4. Create an edge for the running light that is called at 10 Hz.
5. Pass this edge to the block using the input parameter provided for that purpose.
6. Note that the running light always starts with bit 0 and the active bit is reset after the step is exited.
7. Compile, download and test your program changes.

7.7. Exercise 3: Addition of 2nd sequence



New sequence

You can create any number of sequences in one step sequence block (FB with "GRAPH" programming language).

These are inserted graphically in the same window using the "Insert sequence" command. The new sequence appears in the navigation area, depending on the insertion location. A step sequence block with an infinite number of sequences and its associated instance data block represents a sequence complex. All sequences assembled in a function block have a common operating mode and common functions that are blindingly active for all integrated sequences.

Task

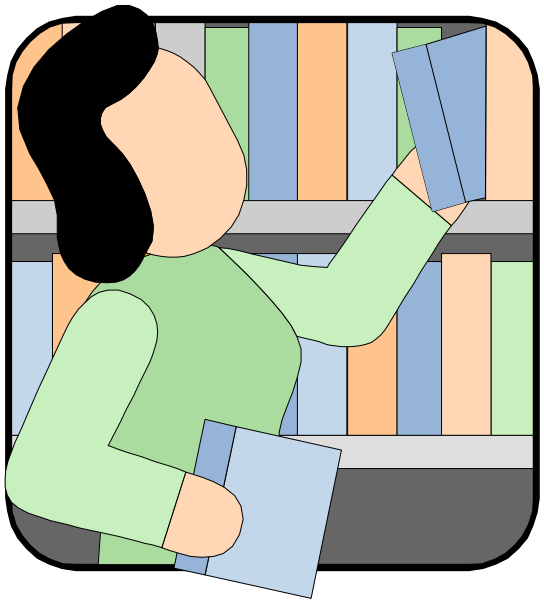
At the end of the sequence section for an invalid weight measurement in the "Removal" step, a jump to a second sequence is to occur. The second sequence transports the workpiece towards the left in the direction of Initiator1 for an approximate duration of 7 s. However, this requires the user to examine the blank once again and to confirm the evacuation to Initiator1 with the pushbutton at bay LB (%I3.4 – S_BayLB). This is indicated by the flashing of the LED "P_BayLB" (%Q3.4) at 2 Hz.

After the transport operation is finished, there is typically a jump back to the starting position step "PickUp".

Procedure

1. Insert a second step sequence with one step in your sequence block.
2. Define the necessary actions in this step and create the required transitions. Also make the changes in the permanent post-instructions.
3. When sequence 1 is exited in the case of an invalid weight, jump to this new sequence and start the transport operation. Change the sub-sequence of sequence 1 in such a way that it corresponds to the new task (note: empty steps are allowed!).
4. After the transport operation finishes (StepX.T >= T#7s), the jump back to the initial position in the main sequence 1 occurs.
5. Download and test your changes.

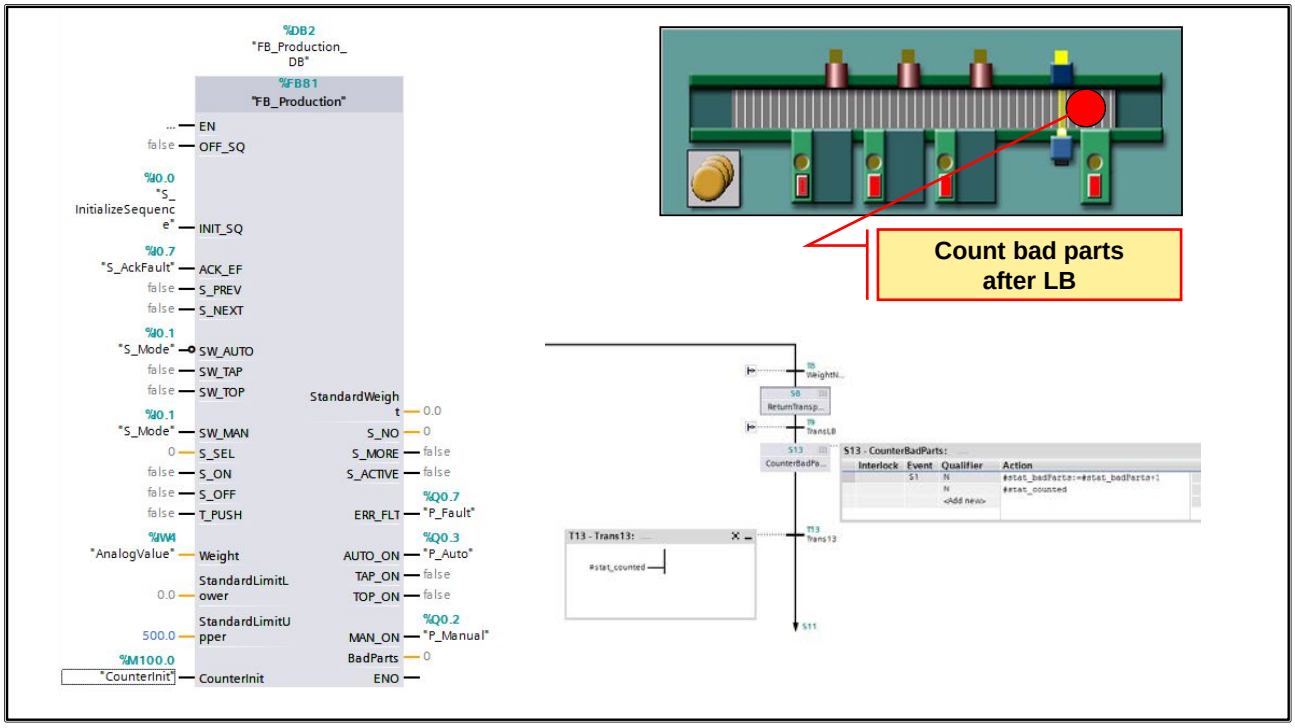
7.8. If you want to know more



Note

The following pages provide additional information or serve as a reference that can be consulted for more complete information on a subject.

7.9. Exercise 4: Count bad parts



Task

The "bad parts" that are separated out (scrap) are to be counted as part of quality control.
You can use a counting operation or an arithmetic operation (addition of a static tag) – with the following advantages:

- A standard counter is not tied up by the sequence.
- More than just 999 scrap parts can be counted.
- The count is always retentive.

The counter value should be returned as a new OUT parameter at the Graph FB. Use a separate IN parameter to initialize the counter. Control the initialization via a new button on the HMI screen or use a watch table.

Procedure

1. If you want to use an arithmetic operation instead of the standard counter, insert another static tag or another output parameter (e.g. "Scrap") in the parameter list.
   If you want to use a standard counter, insert another input parameter of type COUNTER (e.g. "Scrap counter") in the parameter list so that you can assign its number "externally".
   If an IEC timer is used, use the multi-instance function
2. Create another step for counting the bad parts and program the required counting function in this step.
3. Program the initialization of the count value. The initialization must be done differently depending on the counting method.
4. Compile your block again and transfer it with instance DB to the controller. If necessary, check the FB call in OB1. Test whether your changes correspond to the task.

Table of contents

8

8. Operating Mode Management 8-2

8.1. Different operating modes 8-3

8.2. Manual and automatic mode directly in the step 8-5

8.3. Manual and automatic mode with permanent instructions 8-6

8.4. External logic operation in separate block 8-7

8.5. Exercise 1: Operating mode control 8-8

8.6. If you want to know more 8-9

8.7. Exercise 2: Free step selection and activation 8-10

8.8. Example of a possible solution 8-11

8.9. Handling of mode enables 8-12

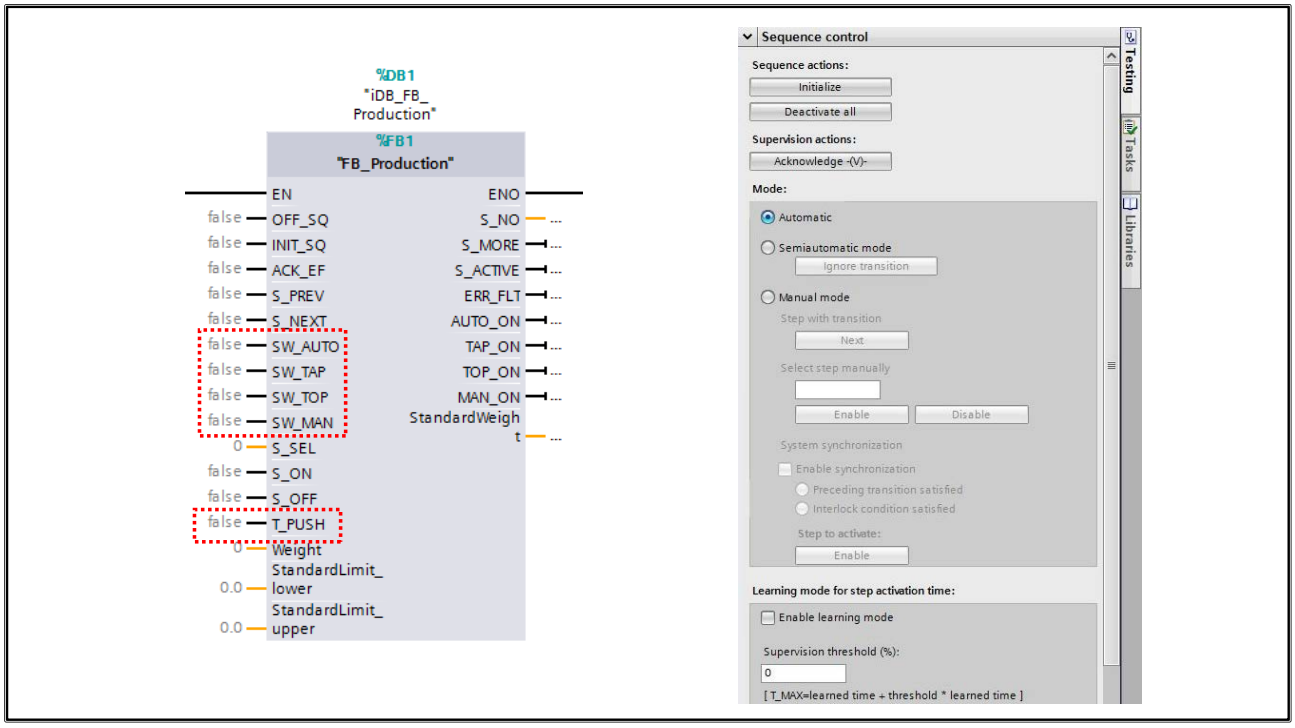
8. Operating Mode Management

The participant should ...

… know and understand the various operating modes

... become acquainted with the application examples for programming and use of these operating modes and initialize the GRAPH FB accordingly

8.1. Different operating modes



General information

The step-enabling behavior of a step sequence is determined by the preselected operating mode. Depending on the parameter assignment, four different operating modes are available:

- SW_AUTO / Automatic mode: When transition condition is true, advancement to the next step occurs.
- SW_TAP / Manual mode – Advancement with transition / transition and advancement:
  In addition to the transition condition being true, a positive edge at FB input "T_PUSH" must be present to advance to the next step.
  (TAP → Transition and Push)
- SW_TOP / Semiautomatic mode / Transition or advancement:
  Advancement to the next step occurs when the condition of the subsequent transition is true or a positive edge is present at FB input "T_PUSH".
  (TOP → Transition or Push)
- SW_MAN / Manual mode: The steps are manually selected and activated or deactivated. When transition condition is true, advancement to the next step does not occur.

Automatic mode

Automatic mode is the default operating mode for GRAPH blocks. This is also the "normal" operating mode after commissioning the block.
To set this operating mode for the GRAPH FB, a positive edge at input "SW_AUTO" is needed.

Manual mode – Advancement with transition

This operating mode enables the step sequence to stay in the step even if the subsequent transition condition is true. The active step is not exited until the transition condition is true and the user gives his "OK". This occurs via the "Next" button in the sequence control or a positive edge at the GRAPH FB input "T_Push".
This operating mode is activated for the FB using input "SW_TAP".

Semiautomatic mode

In this operating mode, advancement to the next step can occur in two ways:

- The condition of the subsequent transition is true and the step sequence automatically activates the next step.
- The user presses the "Ignore Transition" button in the sequence control or the GRAPH block detects a positive edge at input "T_Push". This operating mode allows the user to easily and quickly jump through the sequence.

If a change to this operating mode is to be made using the GRAPH FB, a positive edge must be present at input "SW_TOP".

Manual mode

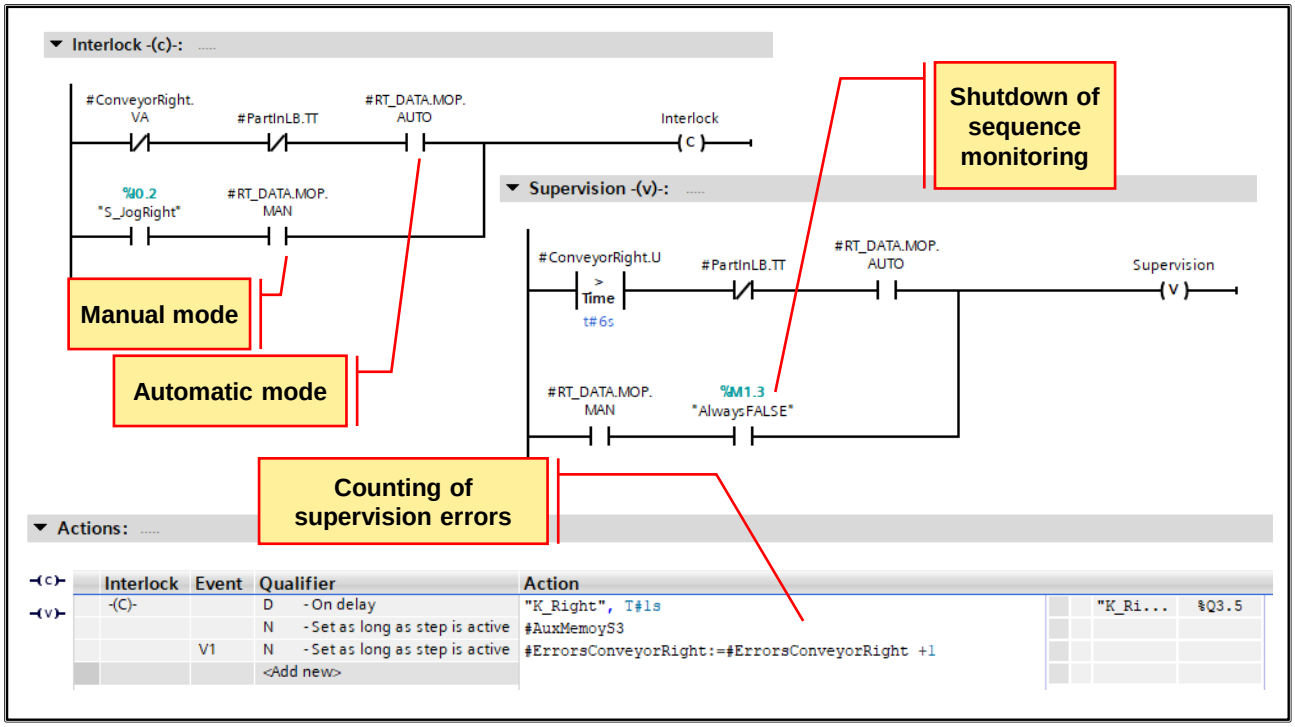
Here, a purely logic operation-oriented control of the plant functions is required since this operating mode is mainly used for setting up the plant and for troubleshooting.

All controls as well as all interlocks/enables must be continuously executed and monitored.

In the individual case, diagnosis and operator input, e.g. on HMI devices, can also occur here.

Manual mode is selected for the GRAPH block using input "SW_MAN".

8.2. Manual and automatic mode directly in the step



General information

The figure shows a simple application for the "#ConveyorRight" step. The conditions for control of the "ConveyorRight" motion are programmed in the interlock logic:

- Automatic mode: "ConveyorRight" until transition "#Part in LB" is true or the supervision becomes active.
- Manual mode: "ConveyorRight" as long as the pushbutton is pressed.

The sequence monitoring is deactivated for manual mode in the logic for the supervision (shutdown of sequence monitoring).

So that the function also works in manual mode, an external step selection must additionally be programmed.

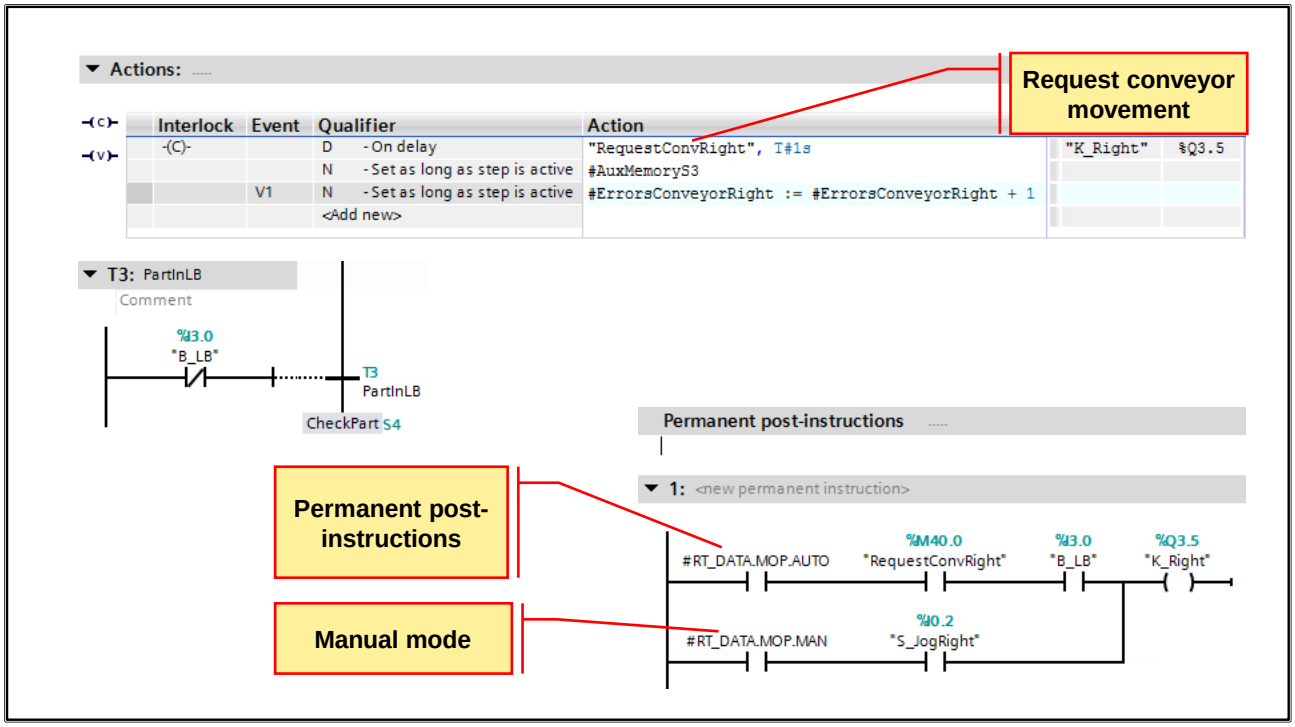
For use of the various operating modes in the GRAPH FB, the internal static parameters of the MOP (mode of operation) section should always be used!

Note

The selection in Manual proves to be more laborious than might appear at first glance. Here, the appropriate step must also be selected and activated for each motion/action.

In many cases, the requires creation of a dedicated function block that, for example, correctly selects the step sequence externally based on its FB parameters via a step assignment list.

8.3. Manual and automatic mode with permanent instructions



Automatic mode

The sequencer provides the motion requests in automatic mode and single step mode (semiautomatic). Request flags must be programmed in the action blocks and, if necessary, step flags (S(i).X) must be used.

These request flags are then further logically combined in the permanent pre- and post-instructions for the motion control.

In the sequencer, only the interlocks in automatic and semiautomatic mode are programmed in the step. The same applies to supervisions and transitions.

Permanent instructions

As shown in the example, the entire control level for the motion is contained in the permanent instructions. This includes, at a minimum, an automatic branch and a manual branch with deactivation of motion when the target is reached (e.g. end position LB1).

Advantages

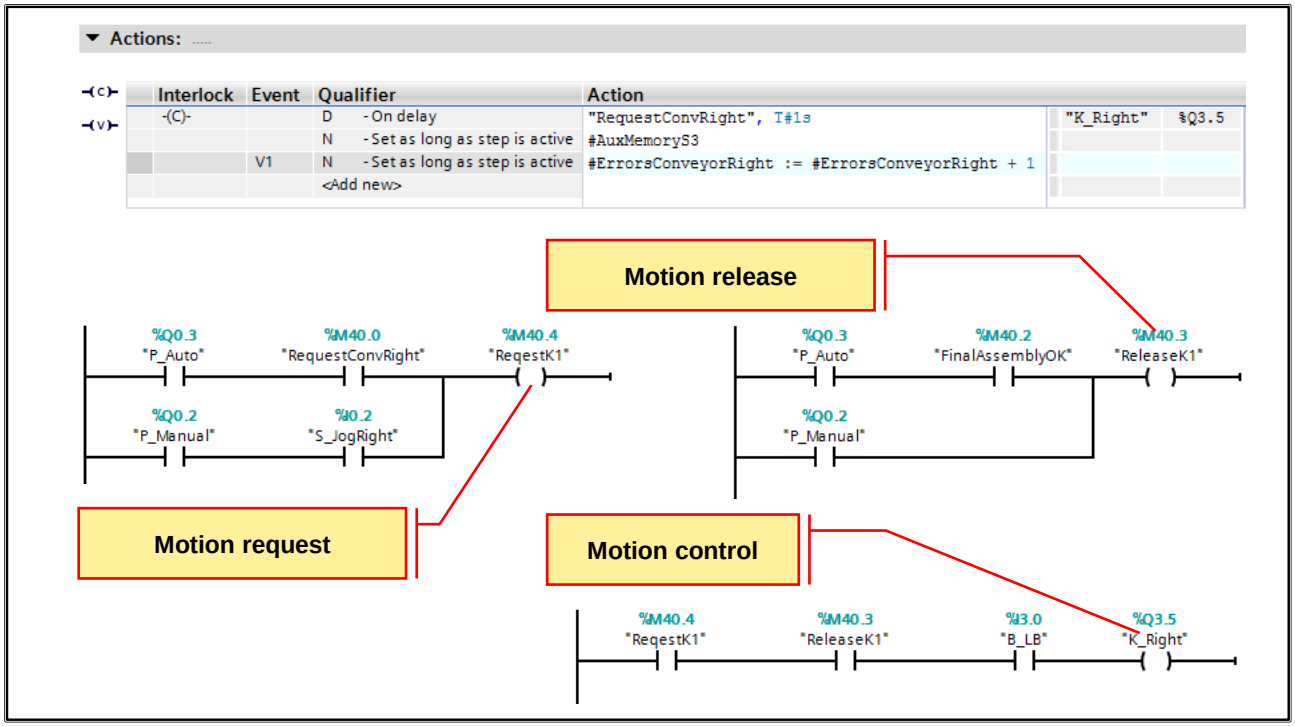
- Integrated in the sequence FB graphically and with documentation.
- Only slightly more memory space and cycle time needed.
- No automatic enable signals necessary in the steps.

Disadvantage

Logic operations in the permanent instructions are not able to undergo diagnostic checks on HMI devices. Detailed diagnostics are thus omitted in manual mode.

This procedure is therefore recommended only for plants with low diagnostic requirements (malfunction lamps + step displays).

8.4. External logic operation in separate block



Automatic mode

The sequencer provides the motion requests in automatic mode and single step mode (semiautomatic). Request flags must be programmed in the action blocks for this.

Manual mode

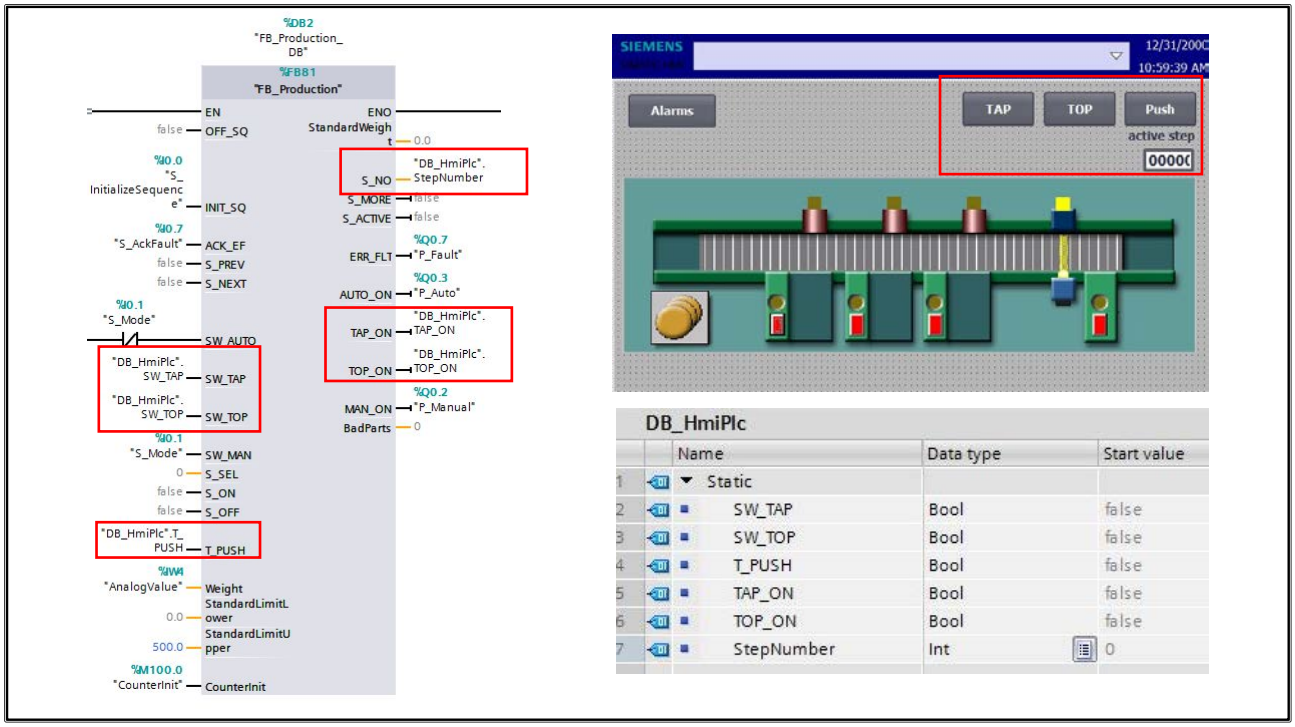
When manual mode is shifted to an external block (FC or FB), the full function scope of STEP 7 is available.

Example

As shown in the figure, there is a classic threefold division for the control of the conveyor motor in automatic mode from step S3:

- Motion request (from the step using bit memory assignment or static FB tag)
- Motion enable using enable signals from the sequence complex (e.g. T003.TT)
- Motion control for the final controlling element "Conveyor_forward" (Q8.5) with inclusion of the signals for the motion request and enable.

8.5. Exercise 1: Operating mode control



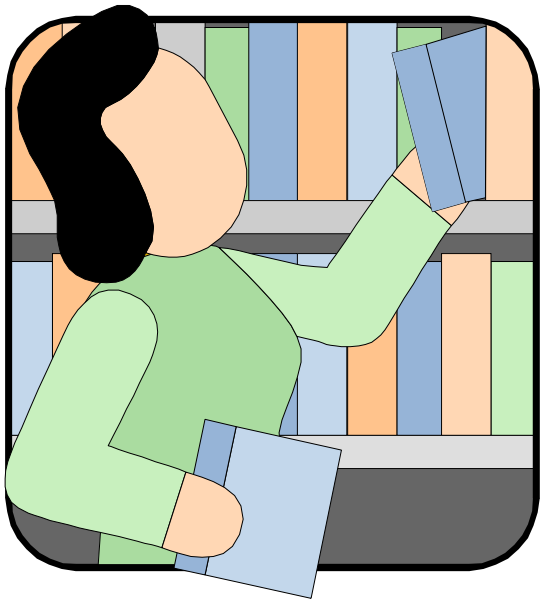
Task

Besides the sequence control in online mode, the step sequence is now also to be switched to "Manual mode – Advancement with transition" and "Semiautomatic mode" via the block interface.

Procedure

1. Add the necessary tags to the "DB_HmiPlc".
2. Connect the tags to the input parameters of the GRAPH FB.
3. Expand the HMI panel for the newly created tags/functions.
4. Compile, download and test the new functions.

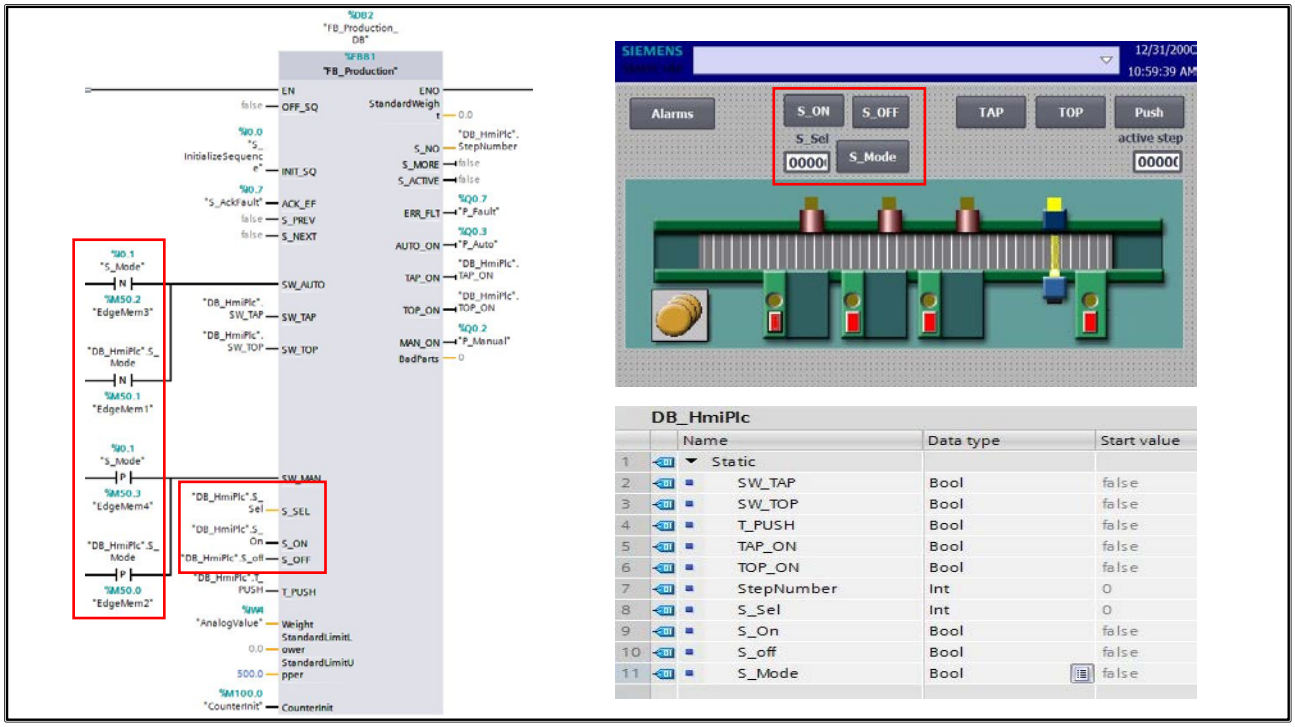
8.6. If you want to know more



Note

The following pages provide additional information or serve as a reference that can be consulted for more complete information on a subject.

8.7. Exercise 2: Free step selection and activation



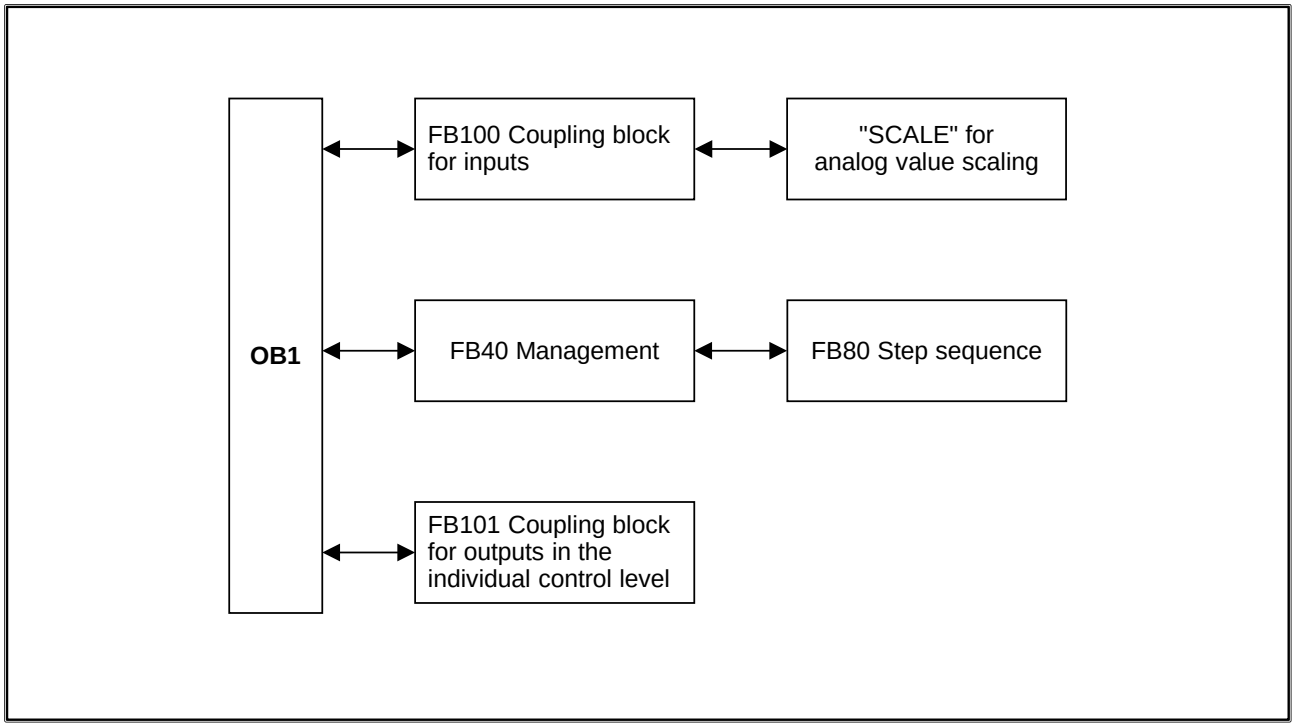
Task

Implement a complete operating mode management via the block interfaces. That is, it should be possible to activate and deactivate steps in manual mode. A free selection of the steps to be activated or deactivated is also to be possible.

Procedure

1. Add the necessary tags to the "DB_HmiPlc".
2. Connect the tags to the input parameters of the GRAPH FB.
3. Expand the HMI panel for the newly created tags/functions.
4. Compile, download and test the new functions.

8.8. Example of a possible solution


OB1
FB100 Coupling block for inputs
"SCALE" for analog value scaling
FB40 Management
FB80 Step sequence
FB101 Coupling block for outputs in the individual control level

Coupling block

The FB 100 is the coupling block for digital and analog inputs. It is needed for complex automation tasks to separate the input addresses from the functionality of the program (e.g. step sequence).
When the program is ported, changes only have to be made in the coupling blocks. This simplifies porting and makes the ported program less prone to errors.

Individual control level

The FB 101 shows the individual control level and the coupling block for the digital outputs. In the individual control level, the digital hardware outputs are assigned dependent on the selected operating mode (manual, automatic, setup, etc.) The states of the outputs are transferred from fixed addresses in the associated instance DB to the respective addresses of the automation system (coupling).

In automatic mode, the current states of the outputs are determined by the actions of the respective step of the step sequence. In manual mode, the current states are overwritten by the manually defined states (e.g. valve open, valve closed).

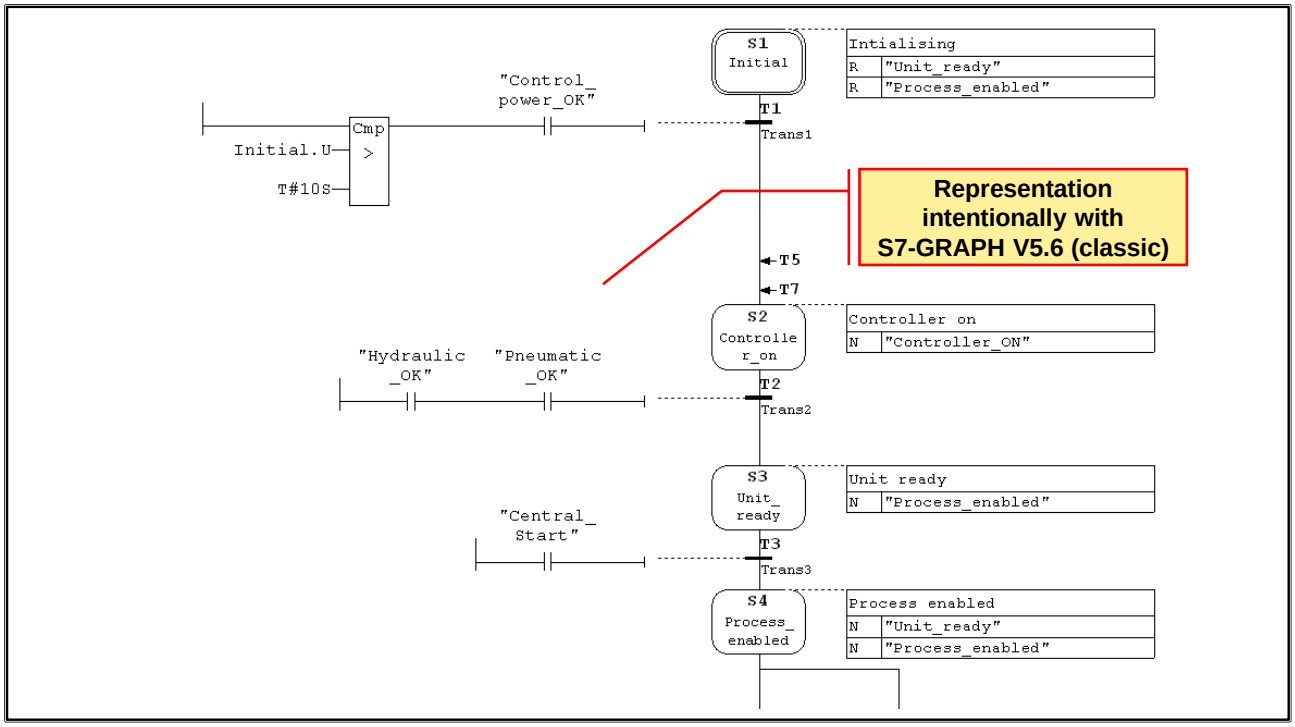
Management

The evaluation of whether or not the step sequence may be triggered occurs in FB 40. Dependent on the result of logic operation, the step sequence block FB 80 is called with the respective instance DB via an interface.

Step sequence

The step sequence is created with GRAPH in the familiar way.

8.9. Handling of mode enables



Mode enables

Production facilities often need media such as hydraulics and pneumatics that the proper functioning of all other units relies on. For this reason it makes sense to have hydraulics and pneumatics switched on from a central place after the plant is switched on and to also generate ready signals for the other blocks. The figure shows an example of how ready signals can be generated in S7-GRAPH.

- After the controller is switched on, the initial step S1 is activated using the "INIT_SQ" parameter and the "Unit_ready" and "Process enabled" signals are reset.
- After a specified minimum wait time, the hydraulic and pneumatic units are switched on with step S2 dependent on the "Control_power_OK" signal with the action "Controller_On".
- After the feedback that hydraulics and pneumatics are OK, the "Process_enabled" alarm is output (to the central controller, etc.) with step S3.
- With the "Central_Start" signal in step S4, the unit is put into operating state and the automatic process is enabled.
- With "Emergency_Off" or loss of the "Door_closed" signal, the process enable is canceled by a change to step S5. If this is acknowledged, a jump to step S2 occurs.
- If an OK signal of the media is lost when the process enable is present in S4, the enable signal is also withdrawn in step S2. As soon as the OK signals of the media are present again, the unit can be put back into the process enabled state with "Central_Start".
- The signals "Controller_On", "Unit_ready" and "Process_enabled" are communicated to the other blocks and the central controller.
- The "Process_enabled" signal enables automatic mode, while the "Unit_ready" signal is used to enable the operating mode enable.

Example of operating mode enable

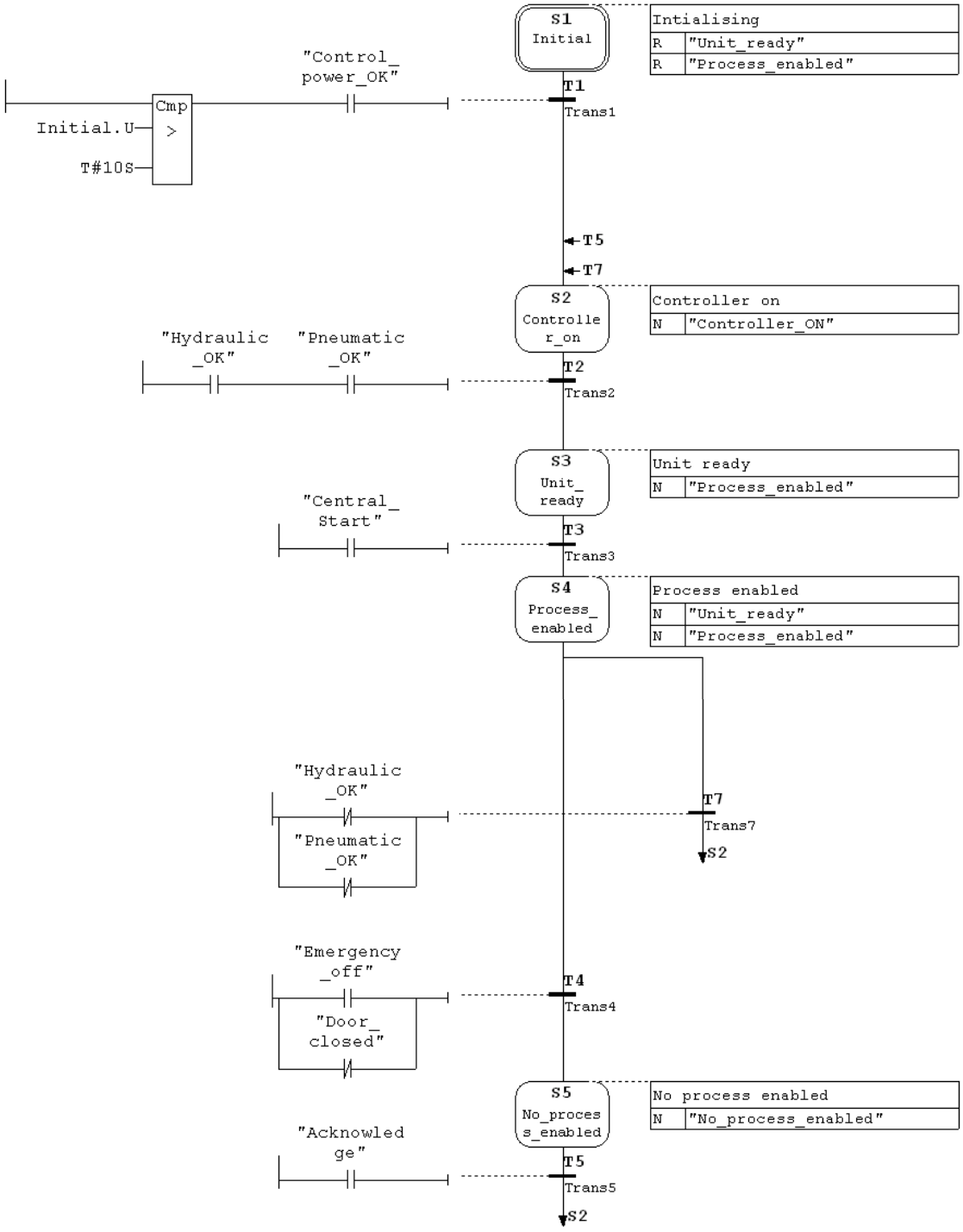


Table of contents

9

9. Troubleshooting and Diagnostics 9-2

9.1. Error categories 9-3

9.2. STEP 7 – Test functions, overview 9-4

9.3. Comparing blocks 9-5

9.3.1. Comparing blocks: Details view 9-6

9.4. Process Diagnostics in WinCC 9-7

9.4.1. Exercise 1: Create HMI screen with PLC Code view and Graph overview 9-8

9. Troubleshooting and Diagnostics

The participant should ...

... be able to analyze a GRAPH sequencer for errors and eliminate these

… be able to apply troubleshooting function for STOP troubleshooting and to use troubleshooting function for error correction

… be able to apply troubleshooting function for locating logic errors and to use troubleshooting function for error correction

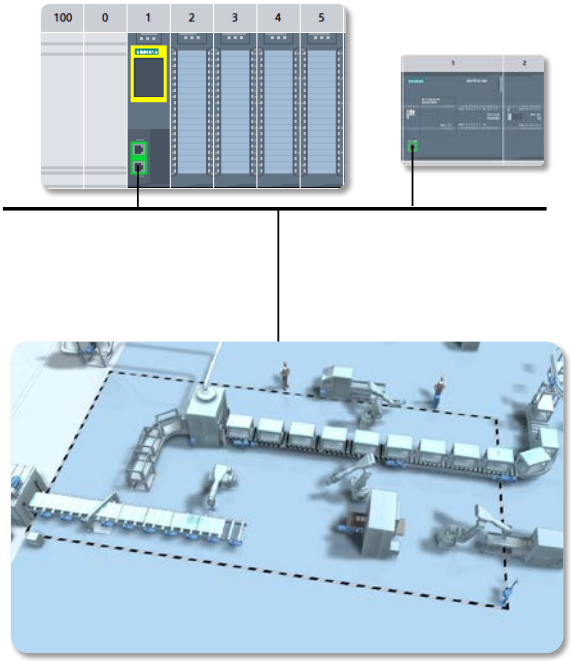
9.1. Error categories

Errors Detected by the System

- Acquiring, evaluating and indicating errors within a PLC
  - Module failure
  - Short-circuit in signal cables
  - Scan time overrun
  - Programming error (accessing a non-existent block)

Functional Errors

- Desired function is either not executed at all or is not correctly executed
  - Process fault (sensor/actuator, cable defective)
  - Logical programming error (not detected during creation and commissioning)



Monitoring functions

Diagnostics is important in the operating phase of a plant or machine. Diagnostics is generally triggered when a fault causes stoppage or malfunction of the plant or machine. Due to the costs associated with downtimes and malfunctions, effort is made to quickly locate and eliminate the cause of the problem.

Error categories

Errors that occur can be divided into two categories, depending on whether or not they are detected by the PLC:

- Errors that are detected by the operating system of the PLC and that normally cause the CPU to go to Stop state.
- Functional errors, i.e. the CPU still processes the programs, but the desired function is not executed at all or is executed incorrectly. Locating such errors generally proves to be much more difficult, because the cause of the malfunction is not known initially.

Possible causes might be:

- A logical programming error (software error) that was not discovered during creation and commissioning of the user program and possibly occurs only under very rare plant conditions.
- Process errors that were triggered by a malfunction of the components involved in the process control, such as cables to sensors/actuators, or by a defect in the sensors/actuators themselves.

9.2. STEP 7 – Test functions, overview

Errors detected by the system
Normal case: CPU in Run

- Programming errors (without OB121): **STOP**
- Access errors (without OB122): **RUN**
- Diagnostic interrupts (without OB82, 83, 86): **RUN**

- **Online & diagnostics**
  - Diagnostics buffer
- **"Testing" task card**
  - Call hierarchy
  - CPU register
  - Local data stack (in planning)
- **Diagnose modules**
  - Diagnostic status (of all modules)

Functional errors
Normal case: CPU in Run

- Process errors (e.g. wire break)
- Logical programming errors (e.g. duplicate assignment)

- **Monitor and modify tags**
  → Monitoring and force tables
- **Monitor blocks (block status)**
  → Monitor in block editor
- **Tools**
  - Cross-references
  - Reference list (I/Q/M/T/C)
  - Call structure

- **"Trace" analyzer function**
- **Program comparison**
- **Set break points (SCL and STL / S7 1500 FW >=2.5)**

Test functions

Various STEP 7 test functions lend themselves to troubleshooting, depending on the nature of the cause of the error:

- **... when CPU in STOP**

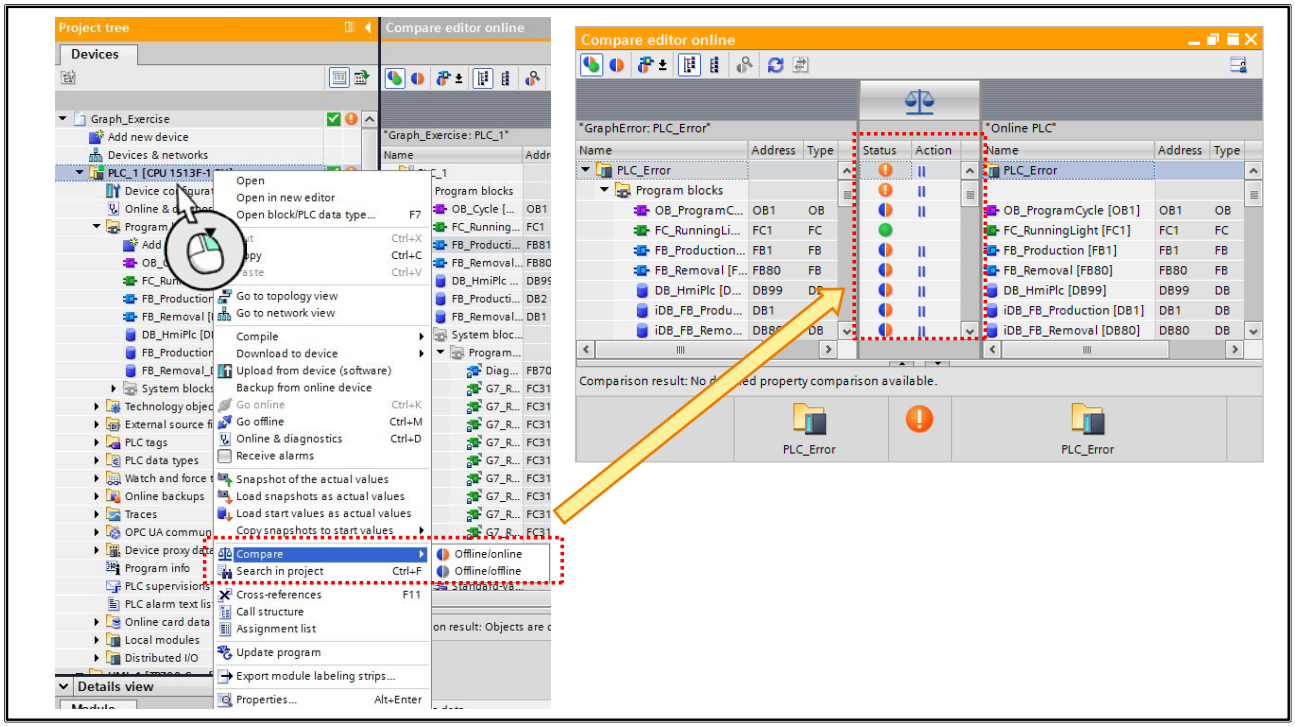
For errors that are detected by the system, the test functions diagnostics buffer, USTACK, BSTACK, LSTACK, and hardware diagnostics provide detailed information about the cause of the error and the interruption point. Through the programming of error OBs (see chapter "Organization blocks"), information about the error can be evaluated via the program and the transition of the CPU to STOP state can be prevented. Because a CPU in STOP state does not read in or output process images and no longer executes the program, use of the test functions Monitor/modify tags and Monitor block are of little use.

- **... when CPU in Run**

Conversely, it is generally of little use to use test functions such as USTACK, BSTACK or LSTACK for troubleshooting when the CPU is in RUN state, because the program execution has not been interrupted and the system provides no information on the error that occurred. The Module information test function can provide only general information about the operating state of the CPU or about errors that occurred in the past. Functional errors can be diagnosed as follows:

- **Process errors** (e.g. wiring error)
  Wiring test of inputs: *Monitor tags*
  Wiring test of outputs: *Enable peripheral outputs* (only in CPU STOP)
- **Logical programming errors** (e.g. duplicate assignment)
  In the search for logical program errors, all listed test functions are used except for *Enable peripheral outputs*.
- **Forcing:** Forcing of operands independent of the program logic
- **Breakpoints:** Program execution in single steps

9.3. Comparing blocks



Comparison types

Two different comparison types are generally available:

Online/offline comparison:

- The objects in the project are compared with the objects of the corresponding device. An online connection to the device is necessary for this comparison.

Offline/offline comparison:

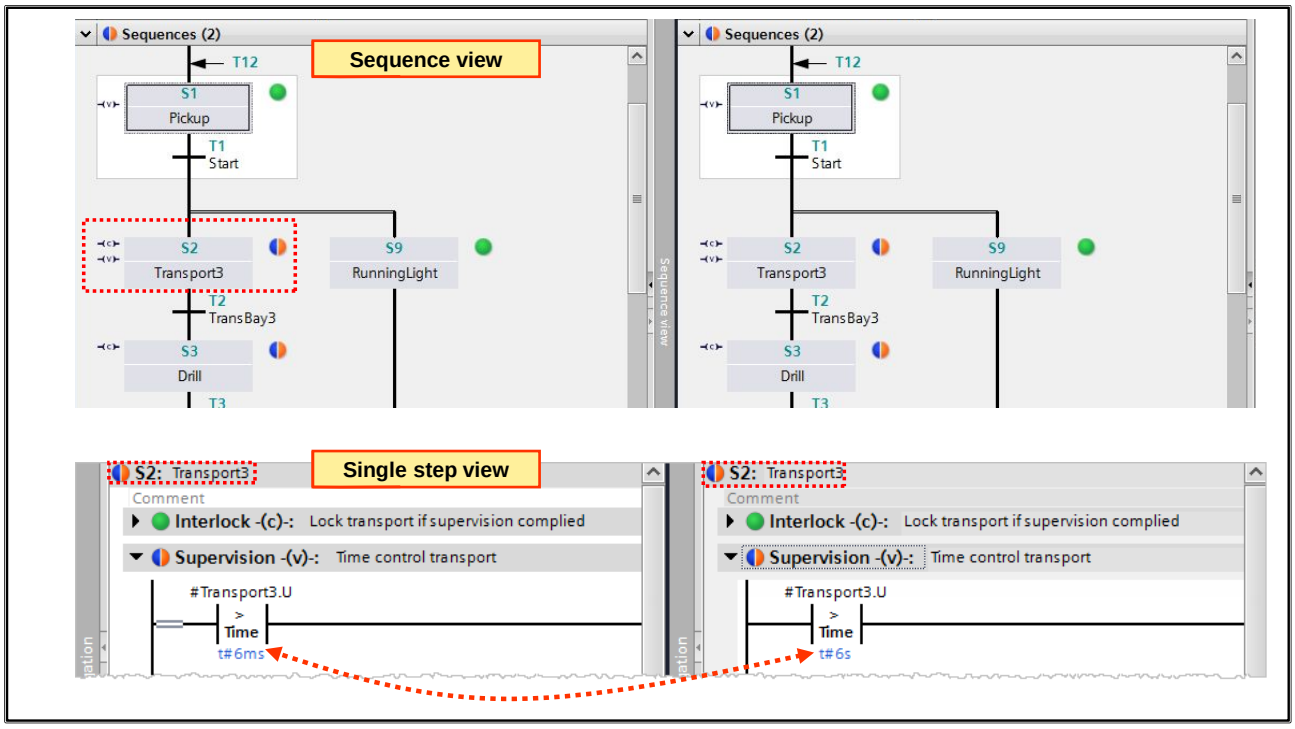
- The objects of two devices either within a project or from different projects are compared using the function.

Symbols of the result display

The results of the comparison are indicated by symbols.
The following table shows the symbols for the comparison results for an online/offline comparison:

Symbol	Meaning
	Folder contains objects whose online and offline versions differ
	Comparison result is not known
	Online and offline versions of the object are identical
	Online and offline versions of the object are different
	Object only exists offline
	Object only exists online

9.3.1. Comparing blocks: Details view



Detailed comparison

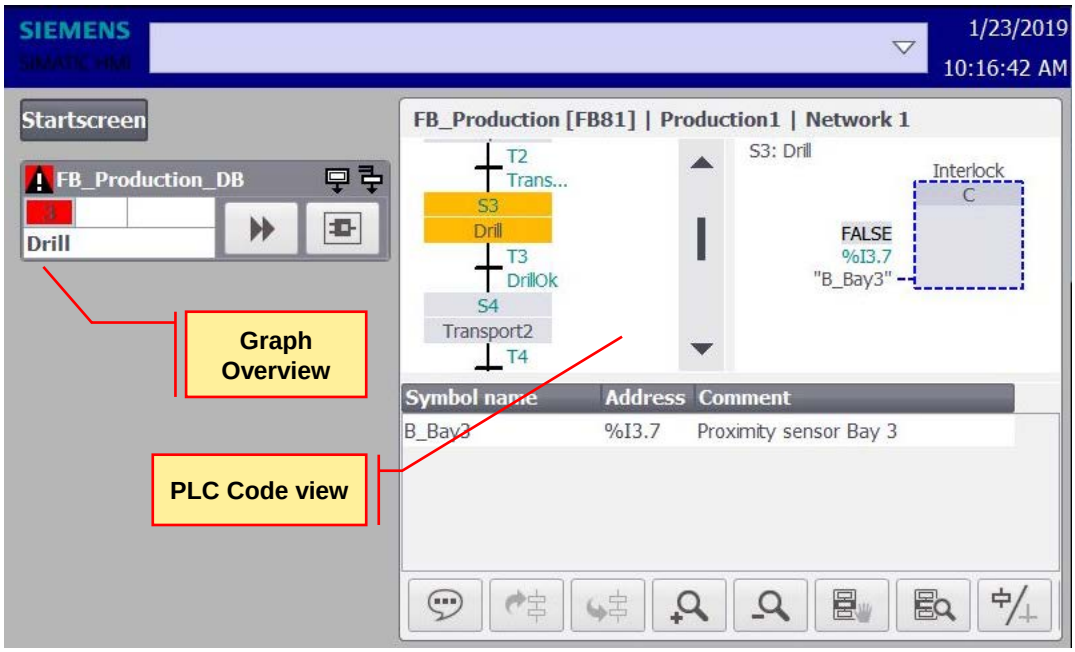
GRAPH blocks can also be checked with a detailed offline/online or offline/offline comparison. The detailed comparison allows you to identify the exact places where the two sequences differ. All views that are possible in the offline configuration are also possible in the detailed comparison.

- Permanent pre-instructions
- Sequence view
- Single step view
- Permanent post-instructions

As in the case of a block comparison, the following symbols provide an overview of the status of the respective element:

Symbol	Meaning
	Online and offline versions of the object are identical
	Online and offline versions of the object are different

9.4. Process Diagnostics in WinCC



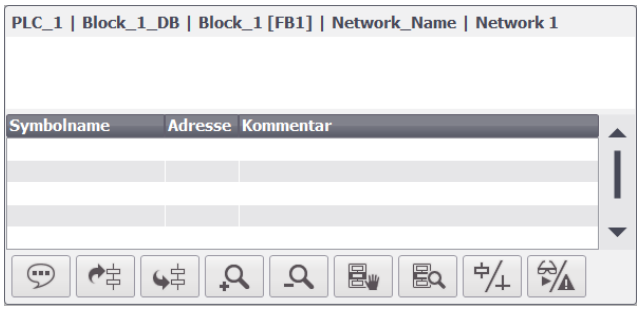
Graph overview and PLC Code view

In order to enable the machine operator to diagnose without a programming device in the event of program errors, various controls are available in WinCC (Comfort / Advanced and Professional).

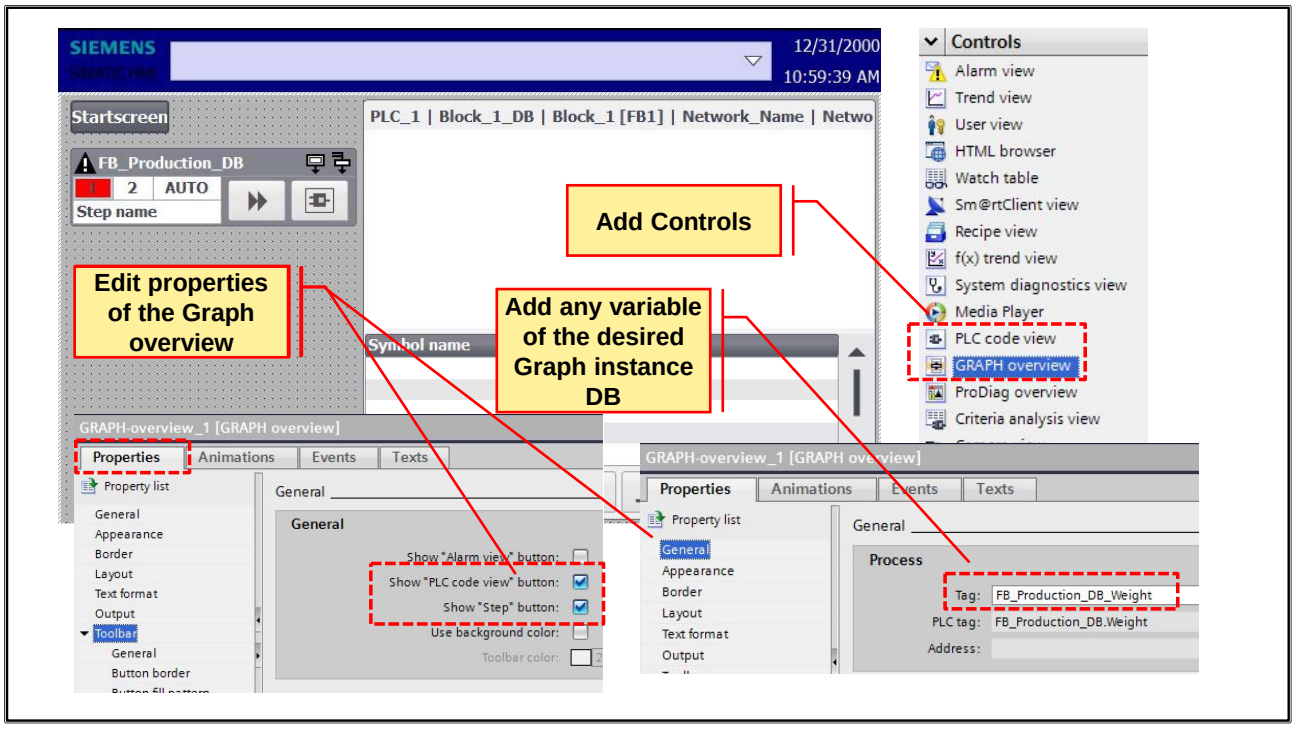
The alarm display has already been explained in the previous chapters.

In the Graph overview you can display the current program status for executed steps.

The PLC code view shows the complete sequencer in online status. The presentation is similar to the usual appearance of the Graph Editor in Step7. In addition to displaying Graph FBs, LAD / FBD blocks are also supported.



9.4.1. Exercise 1: Create HMI screen with PLC Code view and Graph overview



Task

Create a new blank screen in your existing TP 700 Comfort and insert a new graph overview and a PLC code view there. Make sure that your new diagnostic image can be accessed from the startup screen or the Alarms screen

Make the necessary settings and transfer the configuration to the HMI device. Test the functionality by provoking various interferences (interlock, supervision) on the conveyor model.

Note

1. The PLC code display is only actively supplied with data after the operator has pressed the corresponding "PLC code display" button in the Graph overview.
2. In the properties of the Graph overview under "General" a process variable must be connected. In this case, any variable from the corresponding graph instance DB can be assigned. The tag must be visible and accessible from the HMI. If necessary, check the settings in the interface of the FB.

FB_Production

	Name	Data type	Default value	Retain	Accessible from HM..	Wri...	Visible in HMI engineer.
25	<Hinzufügen>				☐	☐	☐
26	Static				☐	☐	☐
27	stat_starting	Bool	false	Non-retain	☑	☑	☑
28	stat_acknowledge	Bool	false	Non-retain	☑	☑	☑
29	stat_removal	Bool	false	Non-retain	☑	☑	☑
30	stat_drilling	Bool	false	Non-retain	☑	☑	☑
31	stat_drillok	Bool	false	Non-retain	☑	☑	☑

Table of contents

10

10. Structure of the Instance Data Block 10-2

10.1. Basic structure of the instance data block 10-2

10.2. Step structure setup 10-3

10.3. Transition structure setup 10-6

10.4. Other static parameters 10-7

10. Structure of the Instance Data Block

10.1. Basic structure of the instance data block

Area	Designation	Length
FB parameters	Name of the input and output parameters of the GRAPH FB	2 bytes: Minimum 10 bytes: Standard/Maximum Customized: User-defined
Reserved work area	G7T_0	16 bytes
Transitions	Transition name (e.g. Trans1)	(Number of transitions) x 16 bytes
Reserved work area	G7S_0	32 bytes
Steps	Step name (e.g. Step1)	(Number of steps) x 16 bytes
Sequence state	Information about sequence state	
Internal work area	Internal data area	

Access

The instance DB can be accessed internally, i.e. within the S7-GRAPH function block, through a logic operation with an "Action" event, for example.

External accesses, access from outside the S7-GRAPH function block, can be realized in the form of a logic operation in another block (OB, FC, FB).

Note

Editing of the instance data block should be avoided. On the other hand, read access to data with STEP 7 tools is permitted.

10.2. Step structure setup

Overview of the static parameters of a step

The following table shows the parameters that are created for each step of the GRAPH function block in the "Static" section of the block interface:

Component	Description	Data type	Minimized DB	Standard DB
S1	Step is activated	BOOL	x	x
L1	Interlock is outgoing	BOOL	x	x
V1	Supervision incoming	BOOL	x	x
R1	Reserved	BOOL	x	x
A1	Error is acknowledged	BOOL	x	x
S0	Step is deactivated	BOOL	x	x
L0	Incoming interlock	BOOL	x	x
V0	Outgoing supervision	BOOL	x	x
X	Step is active	BOOL	x	x
LA	Interlock is unsatisfied	BOOL	x	x
VA	Pending supervision	BOOL	x	x
RA	Reserved	BOOL	x	x
AA	Reserved	BOOL	x	x
SS	System internal	BOOL	x	x
LS	Direct result of the programmed interlock	BOOL	x	x
VS	Direct result of the programmed supervision	BOOL	x	x
SNO	User step number	INT	-	x
T	Total step activation time	TIME	x	x
U	Uninterrupted step activation time	TIME	-	x

T_MAX (S7-1500 only)	Maximum step activation time	TIME	-	x
T_WARN (S7-1500 only)	Warning time	TIME	-	x
CRIT_LOC (S7-300/400 only)	State of the max. 32 LAD/FBD elements in interlock in the current process cycle	DWORD	-	x
CRIT_LOC_ ERR (S7-300/400 only)	Copy of CRIT_LOC, if interlock is outgoing	DWORD	-	x
CRIT_SUP (S7-300/400 only)	State of the max. 32 LAD/FBD elements in supervision in the current process cycle	DWORD	-	x
SM (S7-300/400 only)	System internal	BOOL	-	x
LP (S7-300/400 only)	System internal	BOOL	x	x
LN (S7-300/400 only)	System internal	BOOL	x	x
VP (S7-300/400 only)	System internal	BOOL	x	x
VN (S7-300/400 only)	System internal	BOOL	x	x
H_IL_ERR	System internal	BYTE	-	x
H_SV_FLT	System internal	BYTE	-	x
RESERVED (S7-300/400 only)	Reserve	DWORD	-	x

Access to components

Write access is not recommended because the parameter values influence the sequence. Reading access takes place via the step name followed by the structure element name. In conditions or actions, e.g. the following accesses are thereby possible:

- TROCKNEN.X
- TROCKNEN.T

10.3. Transition structure setup

Transition structure in the interface

The following table shows the parameters that are created for each transition of the GRAPH function block in the "Static" section of the block interface:

Component	Description	Data type	Minimized DB	Standard DB
TV	Transition is valid	BOOL	x	x
TT	Transition is satisfied	BOOL	x	x
TS	Transition is switching	BOOL	x	x
CF_IV (S7-300/400 only)	Entry "CRIT_FLT" is invalid	BOOL		x
TNO	Display of the user-defined transition number	INT		x
CRIT (S7-300/400 only)	State of the max. 32 LAD/FBD transition elements in the current process cycle	DWORD		x
CRIT_OLD (S7-300/400 only)	State of the max. 32 LAD/FBD transition elements in the previous process cycle	DWORD		x
CRIT_FLT (S7-300/400 only)	Copy of "CRIT", if an error occurs	DWORD		x

Access to components

Write access is not recommended because the parameter values influence the sequence. Reading access takes place via the transition name followed by the structure element name. In conditions or actions, e.g. the following accesses are thereby possible:

- VENTIL_EIN.TT
- VENTIL_EIN.CRIT

10.4. Other static parameters

Overview

The static parameters include the step and transition parameters as well as the following information:

- Internal parameters "S_DISPLAY", "S_SEL_OLD", "S_DISPIDX", "T_DISPIDX"
- Component "Operating mode"
- Component "Clock speeds"
- Component "Sequence bit memory"

Note
Additional parameters are contained in a GRAPH-specific instance data block. They are used however only for internal application. The use of these parameters in the program is not recommended.

Internal parameters

The following table shows the internal parameters that are located in the "Static" section. They are used to manage the step and transition numbers:

Component	Description	Data type	Length in bytes	Minimized DB	Standard DB
S_DISPLAY	Copy of the "S_NO" output parameter	INT	2	x	x
S_SEL_OLD	Step number from the "S_SEL" input parameter from the last CPU cycle	INT	2	x	x
S_DISPIDX	Step index for the step in "S_NO" output parameter	BYTE	1	x	x
T_DISPIDX	Transition index for the transition in "T_NO" output parameter	BYTE	1	x	x

Operating mode

The following table shows the "MOP" structure components in the section "Static" that manage the operating modes:

Component	Description	Data type	Minimized DB	Standard DB
AUTO	Switch to automatic mode	BOOL	x	x
MAN	Switch to manual mode	BOOL	x	x
TAP	Switch to "Jog" mode	BOOL	x	x
TOP	Switch to "Automatic" or "Advance" mode	BOOL	x	x
ACK_S	Requirement: Acknowledge step that is indicated in "S_NO" output parameter	BOOL	x	x
REG_S	Requirement: Register step that is indicated in "S_NO" output parameter	BOOL	x	x
T_PREV	Requirement: Output previous valid transition in "T_NO" output parameter	BOOL	x	x
T_NEXT	Requirement: Output next valid transition in "T_NO" output parameter	BOOL	x	x
LOCK	Turn on/turn off interlock handling	BOOL	x	x
SUP	Turn on/turn off supervision handling	BOOL	x	x
ACKREQ	Requirement: Acknowledge step that is indicated in "S_NO" output parameter	BOOL	x	x
SSKIP	Status: "Skip step" is activated	BOOL	x	x
OFF	Deactivate sequencer	BOOL	x	x
INIT	Activate initial steps	BOOL	x	x
HOLD	Stop/resume sequence	BOOL	x	x
TMS_HALT	Stop/resume timers	BOOL	x	x
OPS_ZERO	Switch off/switch on actions	BOOL	x	x

SACT_DISP	Status: Display only active steps	BOOL	x	x
SEF_DISP	Status: Display only steps with errors and faults	BOOL	x	x
SALL_DISP	Status: Display all steps	BOOL	x	x
S_PREV	"Automatic" mode: Display previous simultaneously active step in "S_NO" output parameter "Manual" mode: Display next smaller step number	BOOL	x	x
S_NEXT	"Automatic" mode: Display next simultaneously active step in "S_NO" output parameter "Manual" mode: Display next higher step number	BOOL	x	x
S_SELOK	Requirement: Output step number that is indicated in "S_SEL" input parameter in "S_NO" output parameter	BOOL	x	x
S_ON	"Manual" mode: Activate displayed step	BOOL	x	x
S_OFF	"Manual" mode: Deactivate displayed step	BOOL	x	x
T_PUSH	"Jog" mode: Transition switches, when the satisfied condition is applied with "T_PUSH"	BOOL	x	x
REG	Requirement: Register all interlock and supervision windows	BOOL	x	x
ACK	Acknowledgement of the fault	BOOL	x	x
IL_PERM	Permanent interlock processing	BOOL	x	x
T_PERM	Permanent transition processing	BOOL	x	x
ILP_MAN	Permanent interlock processing in "Manual" mode	BOOL	x	x

With CPUs of the S7-1500 series, the "MOP" structure is located in the "RT_DATA" structure.

Clock pulses (S7-300/400 only)

The following table shows the "TICKS" structure components in the section "Static" that manages the information for activation time calculation. This information can be evaluated only internally in the system.

Component	Description	Data type	Minimized DB	Standard DB
DELTA	Time difference between two cycles	TIME	x	x
OLD	10 ms clock counter value in the last cycle	TIME	x	x
NEW	10 ms clock counter value in this cycle	TIME	x	x

Sequence bit memory

The following table shows the "SQ_FLAGS" structure components in the section "Static" that monitor the internal sequence status:

Component	Description	Data type	Minimized DB	Standard DB
ERR_FLT	General fault	BOOL	x	x
ERROR	Interlock error	BOOL	x	x
FAULT	Supervision error	BOOL	x	x
RT_FAIL	Runtime error	BOOL	x	x
NO_SNO	Selected step does not exist	BOOL	x	x
NF_OFL	Too many steps that have to be activated or deactivated	BOOL	x	x
SA_OFL	Too many active steps	BOOL	x	x
TV_OFL	Too many valid transitions	BOOL	x	x
MSG_OFL	Too many alarms	BOOL	x	x
NO_SWI	Do not switch in this cycle	BOOL	x	x
CYC_OP	Cyclic processing of the sequencer after initialization	BOOL	x	x
AS_MSG	Activate or deactivate alarms during runtime on instruction	BOOL	x	x
AS_SEND	Send alarms from "WR_USMSG" or enter only in diagnostics buffer	BOOL	x	x
SQ_BUSY	Internal edge memory bit for sequence processing	BOOL	x	x
SA_BUSY	Internal edge memory bit for sequence processing	BOOL	x	x
AS_SIG	Edge memory bit for alarms from "Alarm_S" and "Alarm_SQ"	BOOL	x	x

With CPUs of the S7-1500 series, the "SQ_FLAGS" structure is located in the "RT_DATA" structure.

Table of contents 11

11. General Observations about Step Sequences 11-2

11.1. Two types of sequential controls 11-3
11.2. Process-dependent sequential control 11-4
11.3. Motion sequences and signal states 11-5
11.4. Description of control task 11-6
11.5. Written in chronological order 11-7
11.6. Tabular form 11-8
11.7. Motion diagram 11-9
11.8. Function chart according to IEC 61131-3 11-10

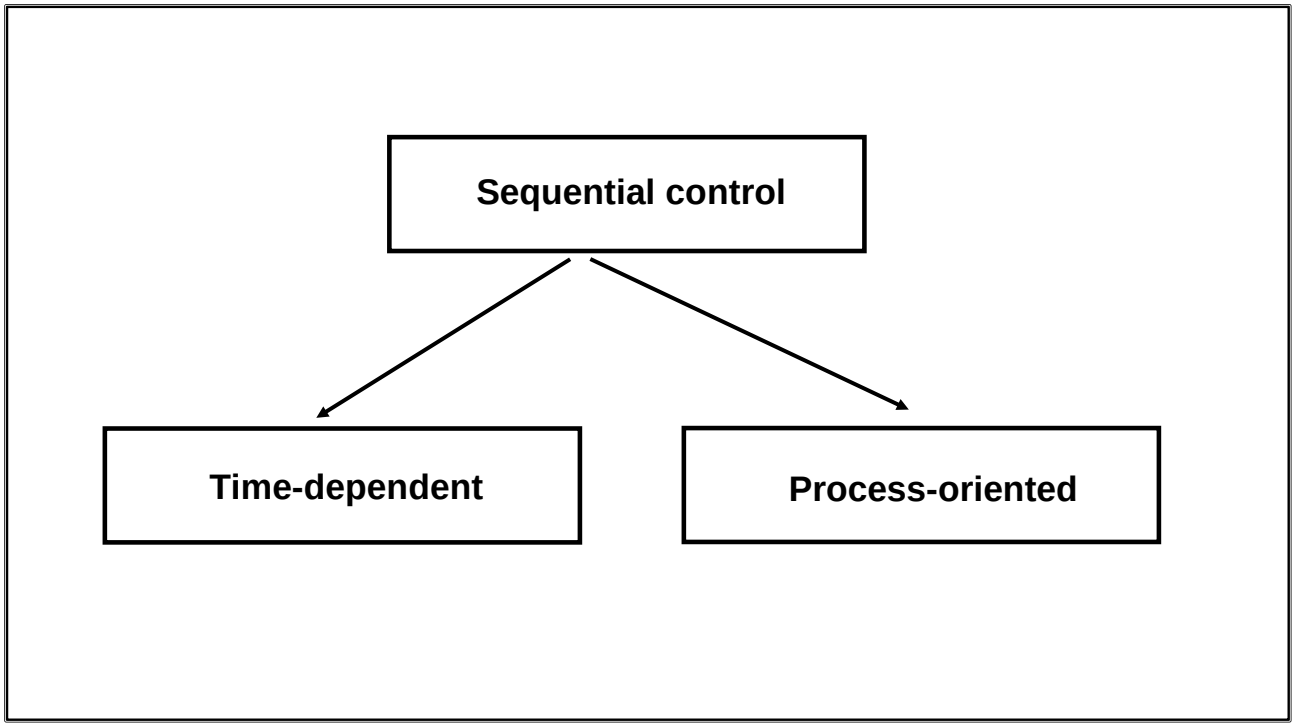
11. General Observations about Step Sequences

The participant should ...

… know the fundamentals of a step sequence

... know the application for step sequences

11.1. Two types of sequential controls



Time-dependent

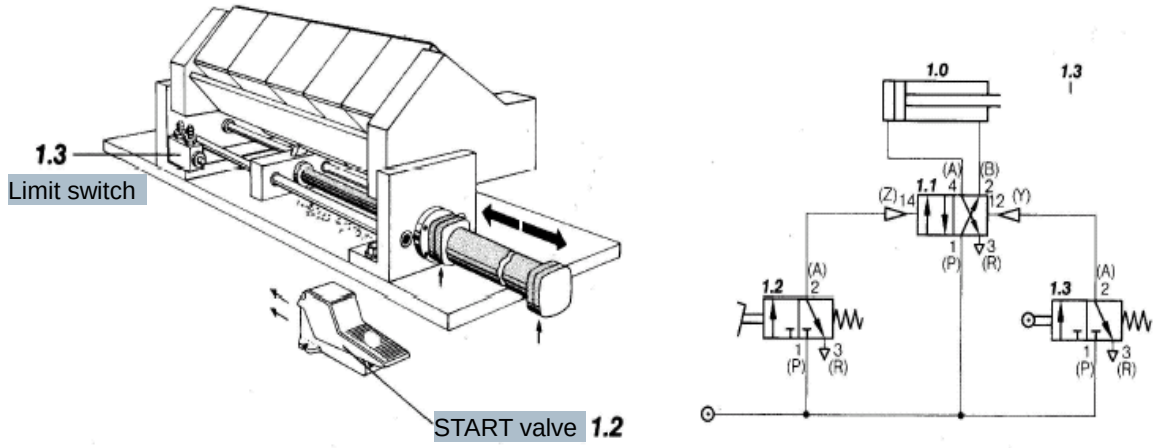
The step-enabling conditions depend on time, e.g. on **wait** or **monitoring** times.

Process-oriented

The step-enabling conditions depend only on signals of the controlled process, e.g. on **feedback signals**.

A combination of time-dependent and process-dependent elements are often found in practice.

11.2. Process-dependent sequential control



Cutting device
When the start valve is actuated, the cylinder of the cutting device extends. After the front end position is reached, the limit switch is actuated and the cylinder automatically retracts again.

General

A distinction is made between logic controllers and sequential controls in control engineering.

Logic controllers

Logic controllers describe the static relationships between input and output signals of a controller.

Sequential control

Control tasks in which the time relationships between inputs and outputs are important are implemented with sequential controls. Another definition for sequential controls is:

"A control with forced stepwise sequence that advances from one step to the next step as programmed, depending on conditions."

Distinguishing features for sequential control are steps and step-enabling conditions: A control task is subdivided into individual steps whose succession is defined by step-enabling conditions. Control commands and step-enabling conditions are assigned to each step.

The step-enabling conditions enable the advancement from one step to the next; the commands of a step consist of control instructions for internal and external units (e.g. set bit memory, start time, switch final controlling element).

11.3. Motion sequences and signal states

- **Description of control task**
- **Written in chronological order**
- **Tabular form**
- **Motion diagram**
- **Function chart**

- S7-GRAPH is a programming language that corresponds basically to a function chart.
- The various representation options will be explained using a program example.

Representation options

The interaction of operation and control elements can be clearly displayed through suitable representation options. The relationships can still be quickly and reliably identified even for demanding tasks.
In addition, a simple representation of motion sequences and switching states enables comprehension by a wide range of experts on a large scale.

Description of control task

The control sequence will be described in text form.

Written in chronological order

The motion sequence is represented in brief lines.

Tabular form

The stepwise sequence is entered in a table.

Abbreviated form

The sequence of motions can be quickly and easily represented by a simplified representation.

Motion diagram

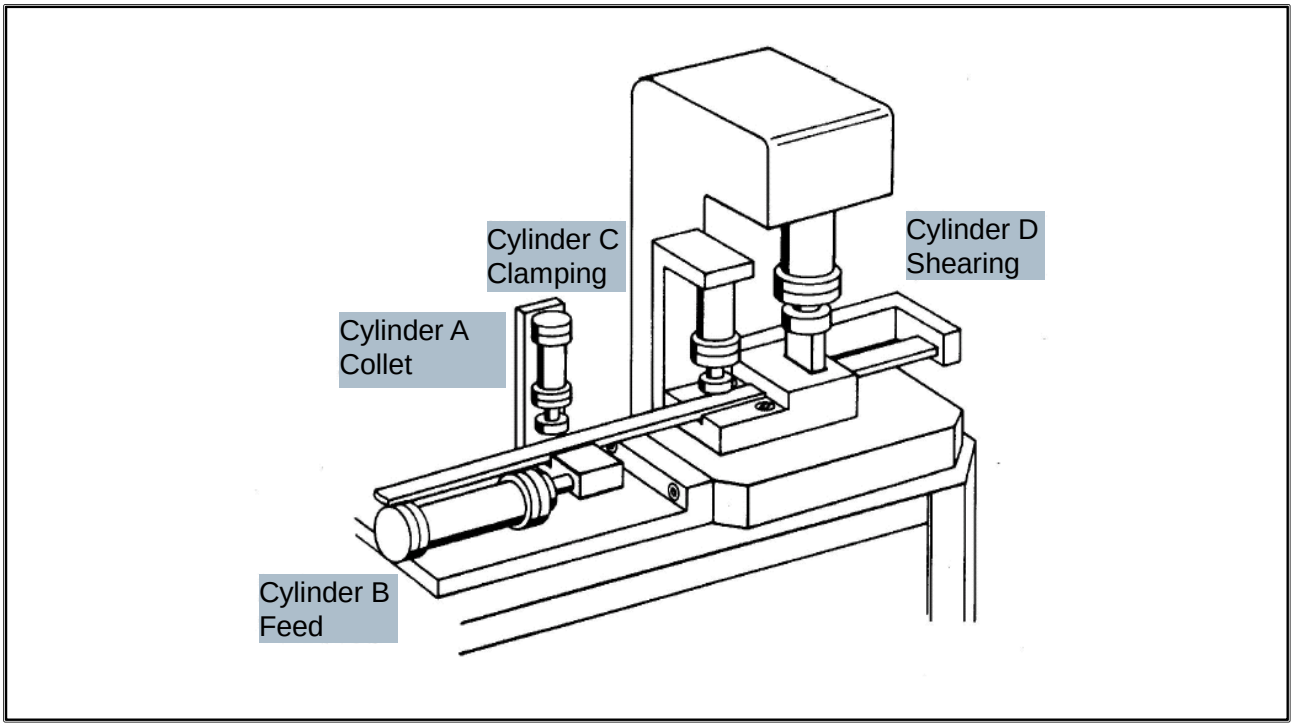
With the help of displacement-step or displacement-time diagrams, the motion sequence is graphically represented. A better overview of the relationships is also provided.

Function chart

A process-oriented representation of the control task. The function chart replaces or supplements the verbal description and represents a control task with its essential properties and the respective applications in a clear way.

S7-GRAPH is a programming language that corresponds basically to a function chart.
The various representation options will be explained using a program example.

11.4. Description of control task



Task

A control for a shearing device is to be designed. Bar material is to be sheared off through the interaction of a feed unit and a shearing device.

- The feed cylinder (cylinder B) performs the feed, The forward and backward stroke of the feed cylinder also moves the pneumatic collet (cylinder A).
- When the material contacts the fixed stop, it is held in place by the clamping device (cylinder C).
- The shearing operation can now begin (cylinder D) and the opening of the collet (cylinder A) occurs simultaneously.
- When the collet (cylinder A) is open, the return to initial position (cylinder B) occurs.
- When the shearing operation is finished (cylinder D) and the feed unit has reached the initial position, the clamping device (cylinder C) is opened and a new sequence of operations can begin.
- The start is initiated by actuating the Start key when all cylinders are in the back end position.

11.5. Written in chronological order

1. Cylinder A extends and closes the collet
2. Cylinder B extends and pushes the material to the endstop
3. Cylinder C extends and clamps the bar material in the shearing device
4. Cylinder A retracts (collet is opened) and Cylinder D extends (shearing)
5. Cylinder B retracts (feed unit moves back) and Cylinder D retracts
6. Cylinder C retracts and opens the clamping device

Written in chronological order

1. Cylinder A extends and closes the collet
2. Cylinder B extends and pushes the material to the endstop
3. Cylinder C extends and clamps the bar material in the shearing device
4. Cylinder A retracts (collet is opened) and Cylinder D extends (shearing)
5. Cylinder B retracts (feed unit moves back) and Cylinder D retracts
6. Cylinder C retracts and opens the clamping device

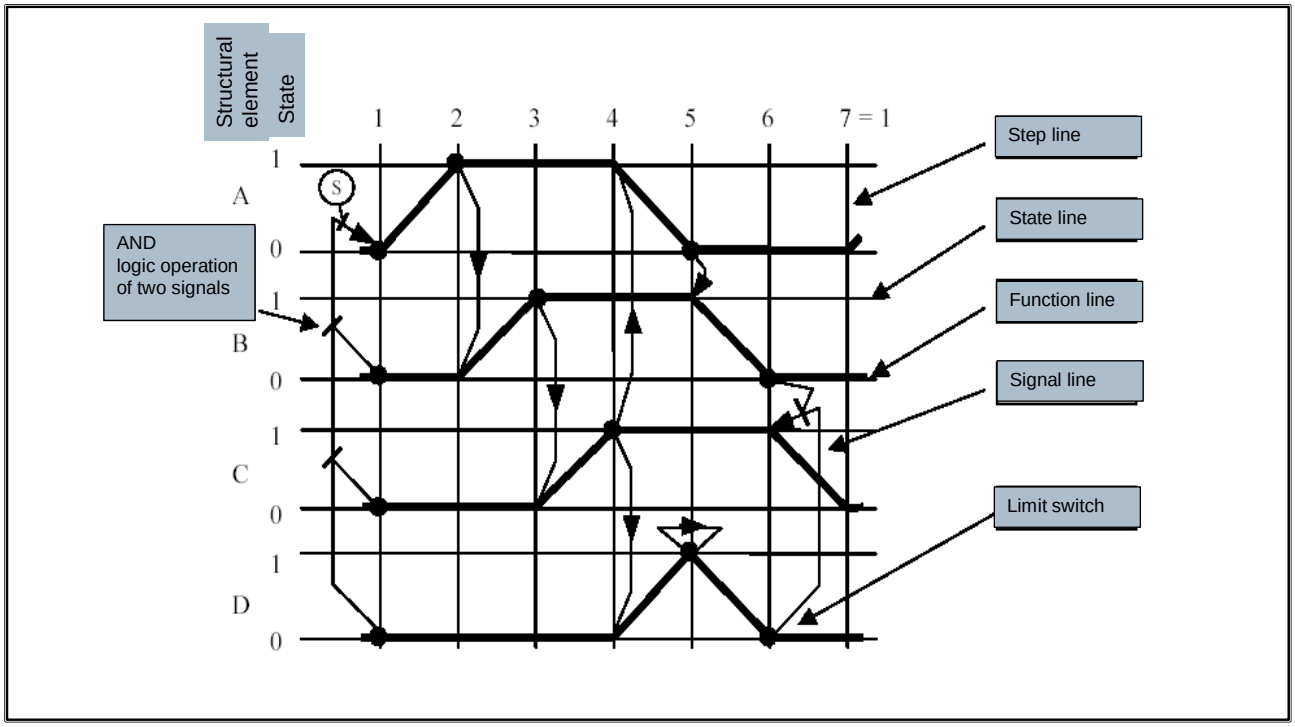
11.6. Tabular form

Step	Cylinder A	Cylinder B	Cylinder C	Cylinder D
1	forward	-	-	-
2	-	forward	-	-
3	-	-	forward	-
4	backward	-	-	forward
5	-	backward	-	backward
6	-	-	backward	-

Tabular form

Representation of the relevant motions in a matrix (tabular form).

11.7. Motion diagram



Displacement-step diagram

The sequence of operations of an operation element is represented here and, namely, the distance traveled is applied as a function of the respective steps (step: change of state of any structural unit).

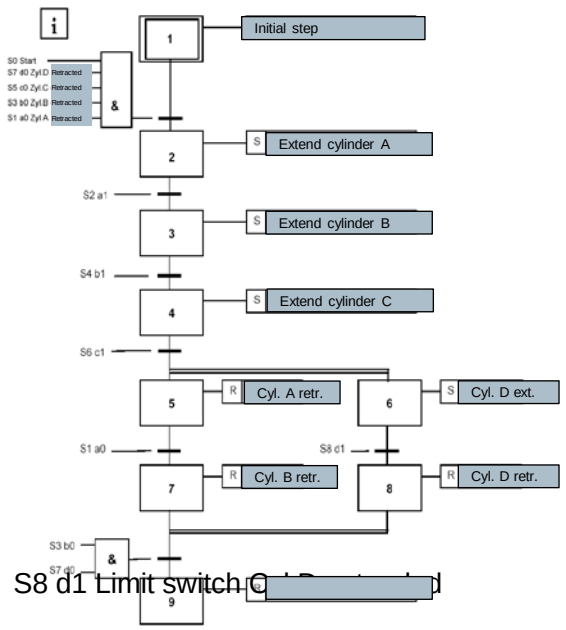
If multiple operation elements are present for a control, these are represented in the same way and drawn one below the other. The relationship is established by the steps.

In the displacement-step diagram, the spacing of the step lines is always the same.
In addition, the signal lines can be included in the displacement-step diagram. The displacement-step diagram looks like this for our program example.

Displacement-time diagram

The displacement-time diagram is essentially a displacement-step diagram in which the time lapse of the motions is also shown. Through a time bar at the lower end of the diagram, the duration of a motion can be read off. The spacing of the step lines varies depending on the time required. The number of steps and the type of motions remain unchanged.

11.8. Function chart according to IEC 61131-3



S0 Start button

S1 a0 Limit switch Cyl.A retracted

S2 a1 Limit switch Cyl.A extended

S3 b0 Limit switch Cyl.B retracted

S4 b1 Limit switch Cyl.B extended

S5 c0 Limit switch Cyl.C retracted

S6 c1 Limit switch Cyl.C extended

S7 d0 Limit switch Cyl.D retracted

Function chart according to DIN EN 61131-3 (IEC 61131-3)

The function chart is a process-oriented representation of a control task, independent of its implementation, e.g. the equipment used.

It facilitates the interaction of various specialties, such as mechanics, pneumatics, hydraulics, process engineering, electrics, electronics, etc. A control task is represented with its essential properties in a basic structure (step field) and with the details required for the respective application in a detailed structure (command field) in a clear way.

Note

Because the shearing device involves two separate workstations (feed station and shearing device), the function chart must be created with a simultaneous branch.

S0 Start button

S1 a0 Limit switch Cyl.A retracted

S2 a1 Limit switch Cyl.A extended

S3 b0 Limit switch Cyl.B retracted

B extended S4 b1 Limit switch Cyl.

S5 c0 Limit switch Cyl.C retracted

S6 c1 Limit switch Cyl.C extended

S7 d0 Limit switch Cyl.D retracted

S8 d1 Limit switch Cyl.D extended

Contents

12

12. Training and Support 12-2

12.1. Any Questions on our Training Courses Offered?? 12-3

12.2. www.siemens.com/sitrain 12-4

12.3. Learning path: SIMATIC S7 Programming in the TIA Portal 12-6

12.4. Download the training documents 12-7

12.5. The Industry Online Support 12-8

12.6. The Principle of Navigation 12-9

12.7. Complete product information 12-10

12.8. mySupport – Overview 12-11

12.9. Support Request 12-12

12.10. Support Request 12-13

12.11. Industry Online Support – wherever you go 12-14

12.11.1. Scanning product / EAN code 12-15

12.11.2. Scan functionality 12-16

12.12. Forum - the communication platform for Siemens Industry products 12-17

12.12.1. Conferences and Forum management 12-17

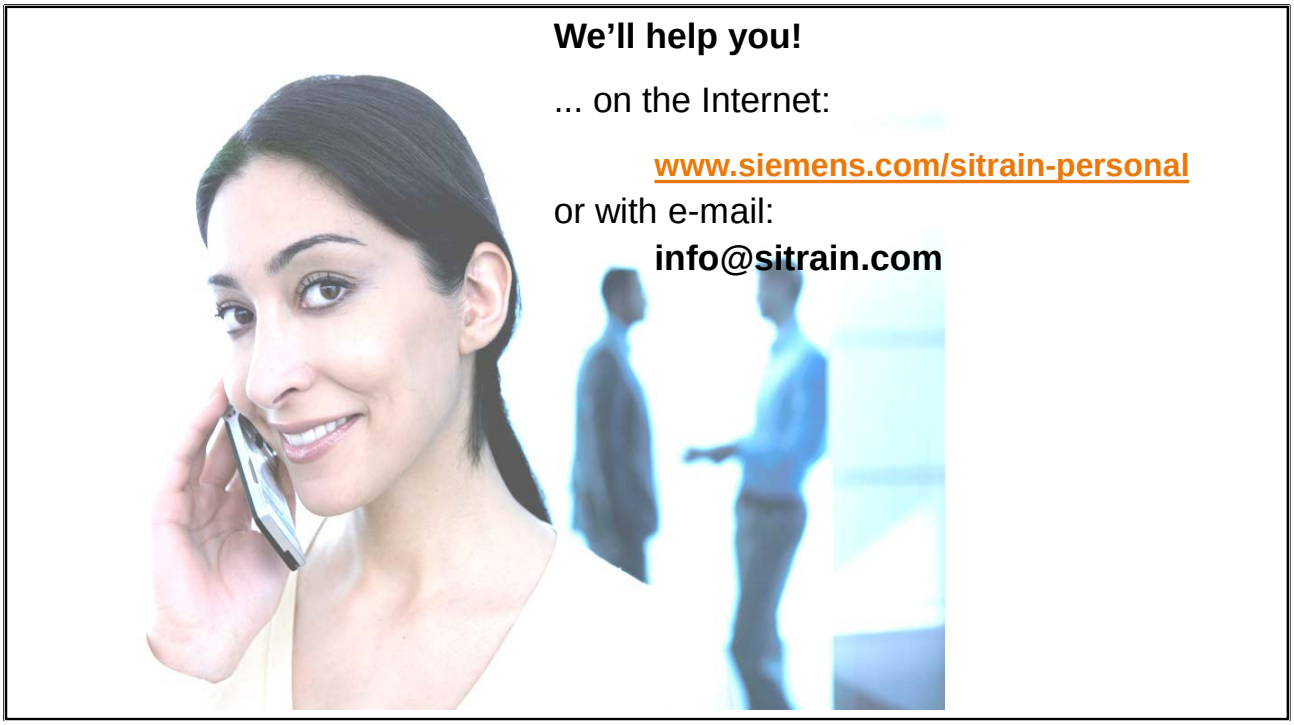
12.12.2. Interactions in the Forum 12-19

12.13. Task and Checkpoint 12-21

– the most important information

12. Training and Support

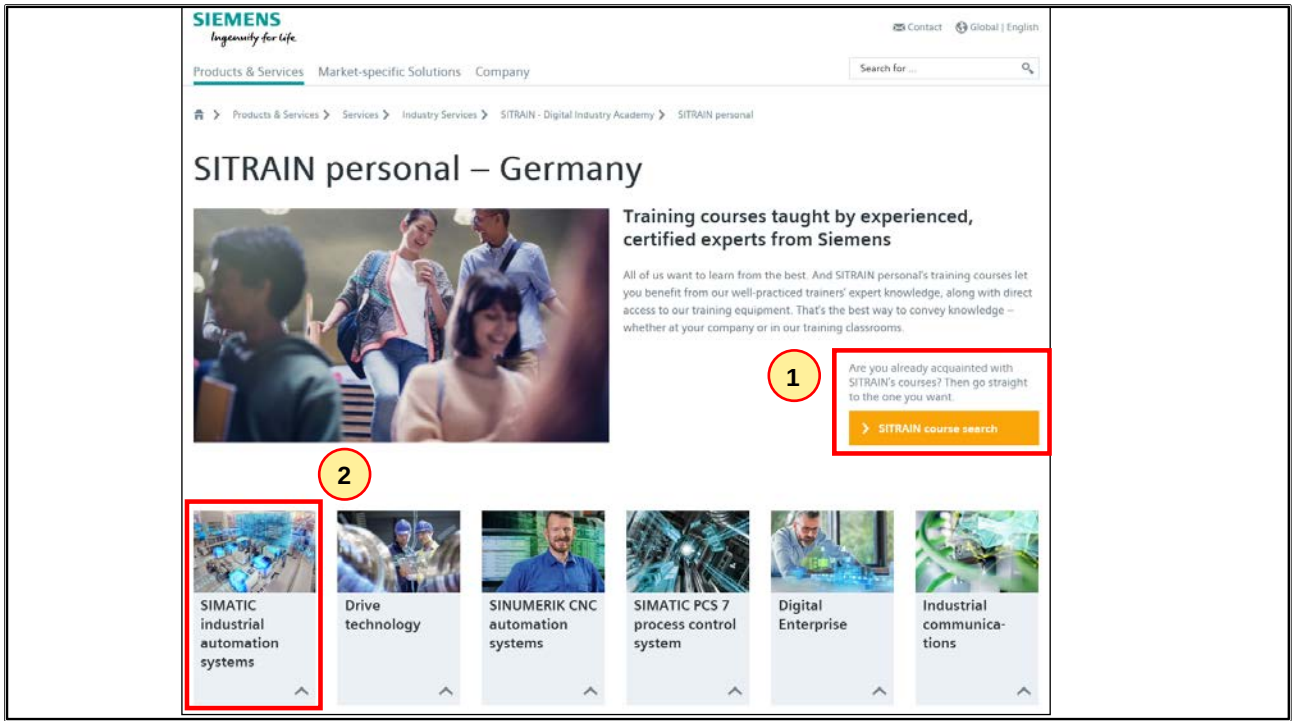
12.1. Any Questions on our Training Courses Offered??



General Information

We'll be glad to help you regarding any questions on our training courses offered.

12.2. www.siemens.com/sitrain


SIEMENS
Ingenuity for life
Contact
Global | English
Products & Services
Market-specific Solutions
Company
Search for ...
Products & Services > Services > Industry Services > SITRAIN - Digital Industry Academy > SITRAIN personal
SITRAIN personal – Germany
Training courses taught by experienced, certified experts from Siemens
All of us want to learn from the best. And SITRAIN personal's training courses let you benefit from our well-practiced trainers' expert knowledge, along with direct access to our training equipment. That's the best way to convey knowledge – whether at your company or in our training classrooms.
1
Are you already acquainted with SITRAIN's courses? Then go straight to the one you want.
SITRAIN course search
2
SIMATIC industrial automation systems
Drive technology
SINUMERIK CNC automation systems
SIMATIC PCS 7 process control system
Digital Enterprise
Industrial communica-tions

The complete range of courses offered can be accessed via the following links:

www.siemens.de/sitrain or

www.siemens.com/sitrain

Course Search

1 The course search permits the user to find the required courses by applying different search filters such as keyword, target group, etc. The filters can also be combined.

Course Catalog

The course catalog permits you to find the required course via learning paths or via the Siemens Mall structure.

2

> Home > Industrial Automation > Automation Systems > SIMATIC Industrial Automation Systems

SIMATIC Industrial Automation Systems

Consistent and efficient

A centerpiece of our comprehensive range of products and services for industrial automation is SIMATIC, a unique, consistent system of first-class products for every field of application, in all industries. Regardless of whether it's manufacturing and process automation or solutions for infrastructure tasks: with SIMATIC we make an important contribution toward improving your productivity.

SITRAIN has a portfolio of training courses that are perfectly matched to your requirements and your plant's lifecycle.

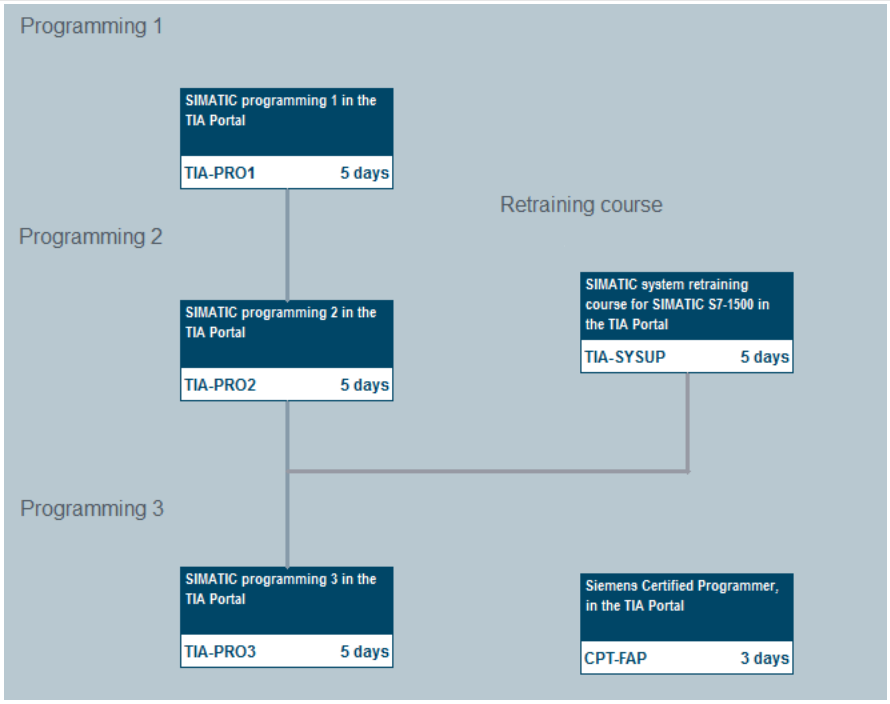
SIMATIC S7 TIA Portal

- On the path to the digital enterprise - discover your potential with training courses for SIMATIC S7-1500 training in the TIA Portal
- SIMATIC TIA Übersicht
- SIMATIC S7 Programming in the TIA Portal
- SIMATIC S7 Service Training in the TIA Portal
- SIMATIC Safety Integrated in the TIA Portal
- SIMATIC S7 Engineering Tools in the TIA Portal
- SIMATIC Technology im TIA Portal
- SIMATIC S7-1200

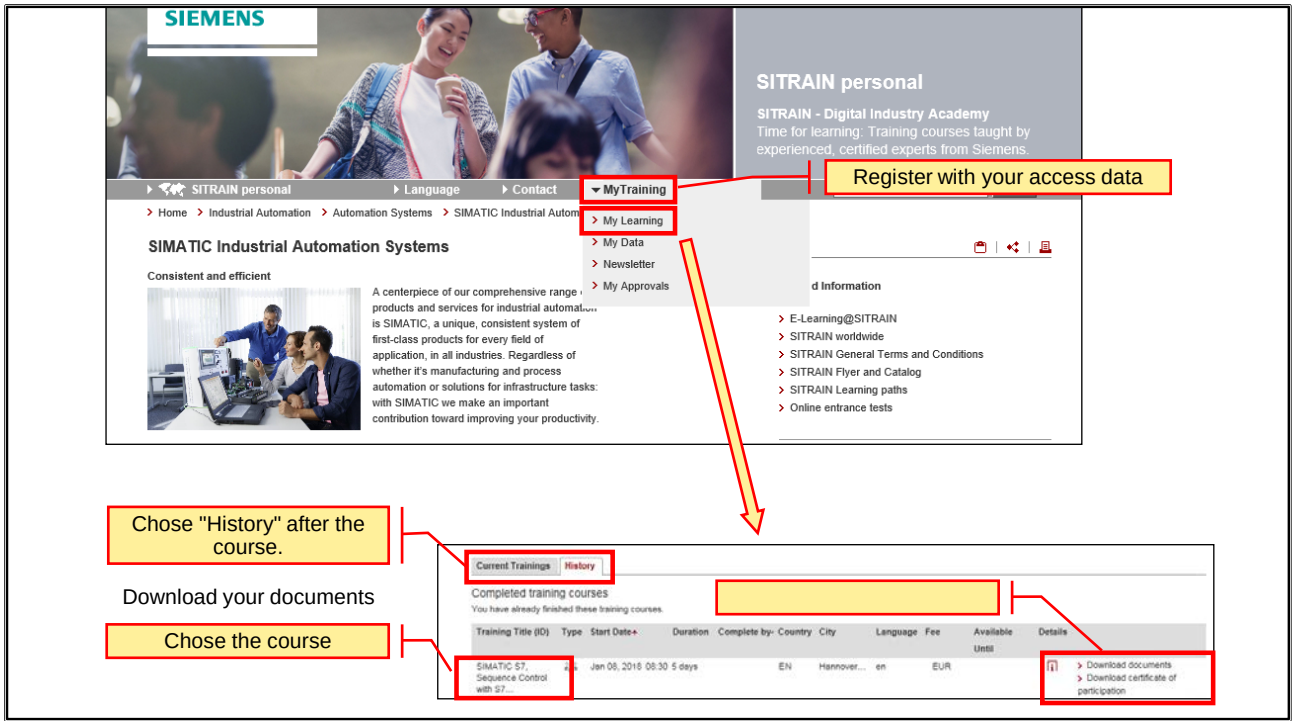
SIMATIC S7-300/-400 with STEP 7 V5.x

- SIMATIC S7 Trainings based on SIMATIC S7-300/-400 with STEP 7 V5.x
- SIMATIC S7 Programming based on STEP 7 V5.x
- SIMATIC S7 Service Training based on STEP 7 V5.x
- SIMATIC Safety Integrated based on STEP 7 V5.x
- SIMATIC S7 Engineering Tools based on STEP 7 V5.x
- SIMATIC Technology based on STEP 7 V5.x

12.3. Learning path: SIMATIC S7 Programming in the TIA Portal

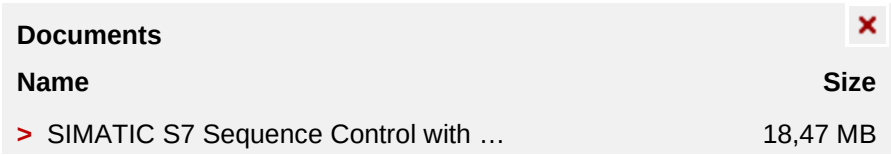

Programming 1
SIMATIC programming 1 in the TIA Portal
TIA-PRO1 5 days
Retraining course
Programming 2
SIMATIC system retraining course for SIMATIC S7-1500 in the TIA Portal
TIA-SYSUP 5 days
SIMATIC programming 2 in the TIA Portal
TIA-PRO2 5 days
Programming 3
SIMATIC programming 3 in the TIA Portal
TIA-PRO3 5 days
Siemens Certified Programmer, in the TIA Portal
CPT-FAP 3 days

12.4. Download the training documents



If you want to download the training documents, proceed as follows:

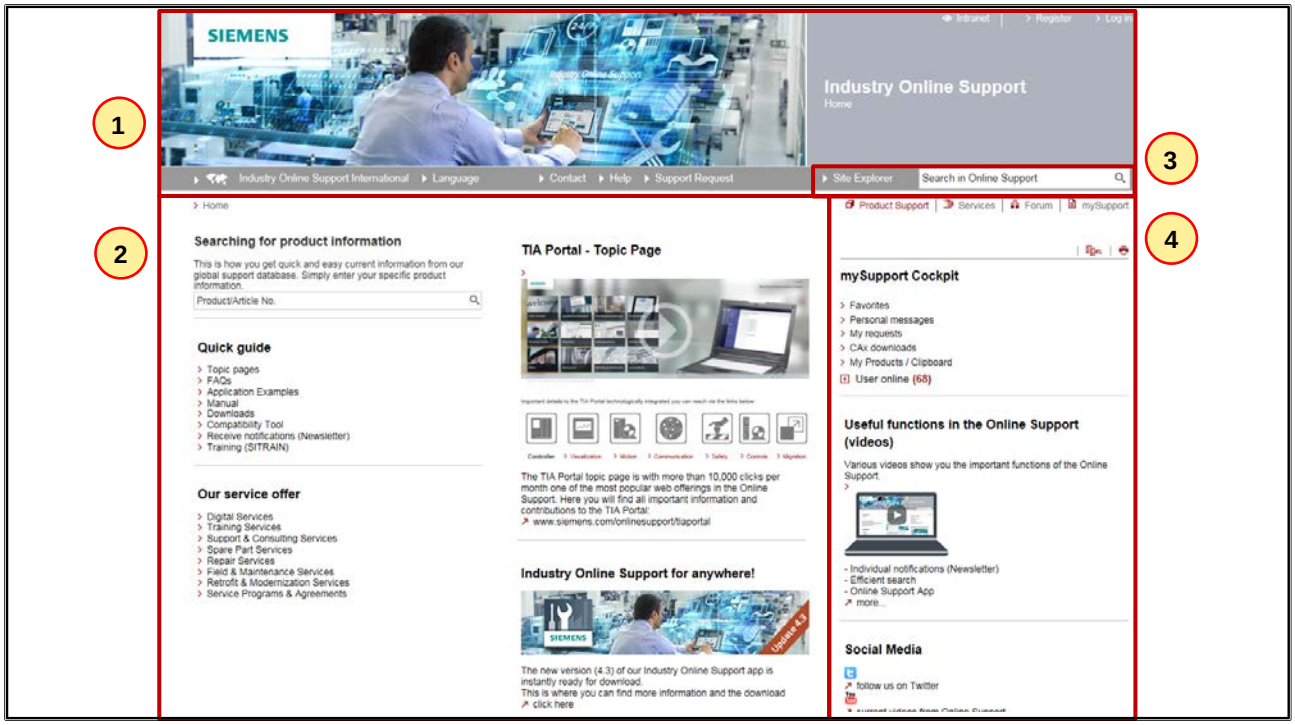
- Visit our new SITRAIN homepage at http://www.siemens.de/sitrain
- Register with your access data under the menu option **MyTraining**.
- Select **MyLearning** on the right-hand side of the submenu.
- Select your course and download your documents with a click on ”Download documents”.



Hint:

Please note that the training documents may be used for personal purposes exclusively. You agree that you will not copy the training documents or make them accessible to third parties and that you will be liable for any damage resulting thereof.

12.5. The Industry Online Support – the most important innovations



The most important functions are always in the same place on all the pages:

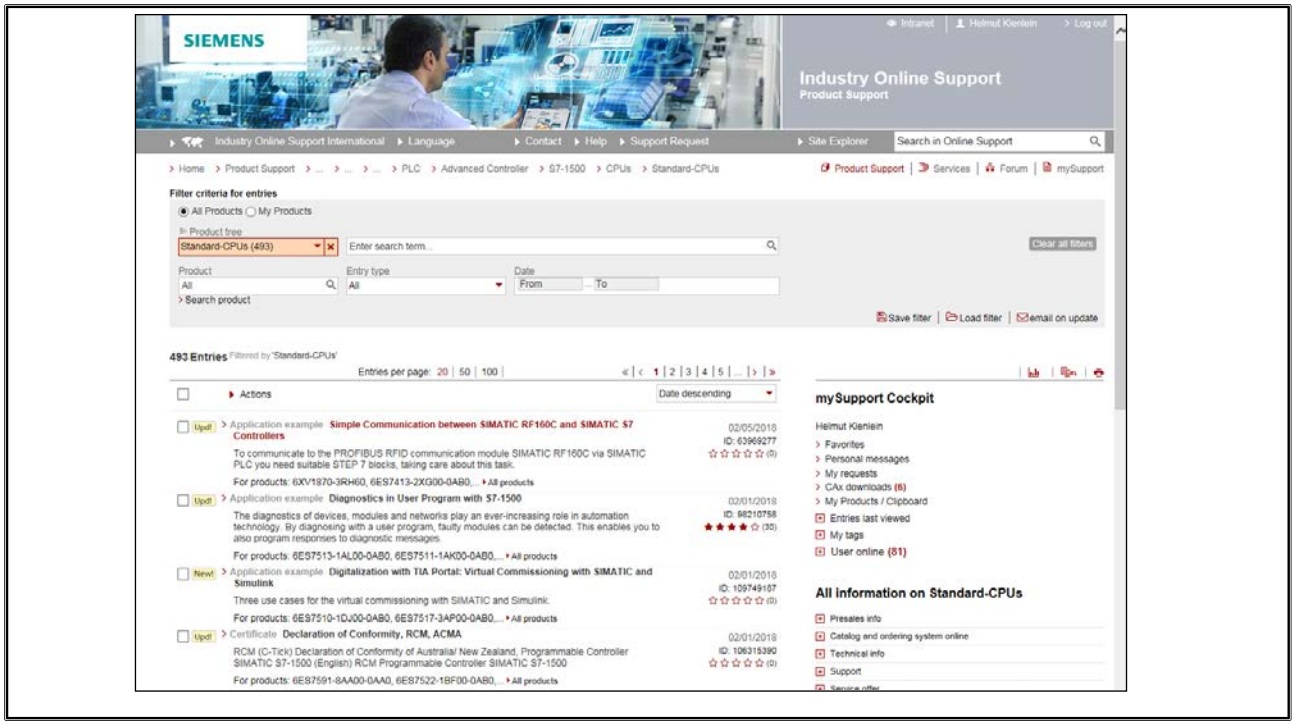
1 The menu bar links to the main areas of the site. You can subscribe and register at any time to benefit from the features the personalized mySupport option offers.

2 Links to our service offerings are in the center. On the start page, you will find up-to-date information and links, which quickly brings you to your destination in other areas of Online Support.

3 Links from the menu bar are repeated at the top of the page: Product Support, Services, Forum and mySupport.

4 On every page, you will find your personal mySupport cockpit. There, for example, you can see when the status of your support inquiry changes.

12.6. The Principle of Navigation



Here, you will find information about all the current and discontinued products, such as:

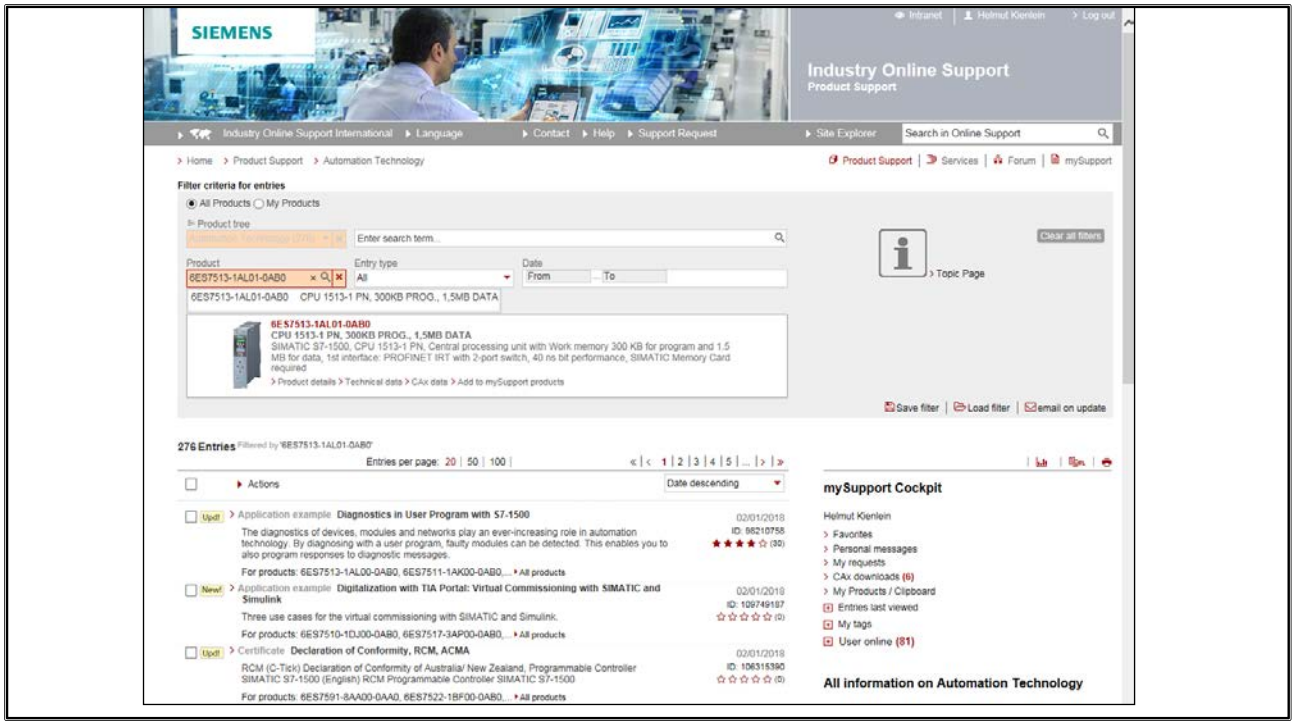
- Frequently Asked Questions (FAQ)
- Manuals and Operating Instructions
- Downloads
- Product Notes (product announcements, discontinuation, etc.)
- Certificates
- Characteristics
- Application Examples

You will not only be able to access these articles though the product tree, but also through a central filter bar. The integration of various search filters will give you access to relevant information after only a few clicks. The product tree has been moved to an equivalent filter. This has the effect that several filter steps can be combined clearly and comprehensibly.

Based on the preview numbers you can see the expected set of results before using a filter. This makes finding relevant information considerably easier and more efficient.

For example, you can customize your search by combining the product tree, a search keyword and a document type in your search.
There will be no hidden search parameters; all the settings and results will be clearly displayed.

12.7. Complete product information



A powerful function of the Industry Online Support is the direct access to complete product information. You can use it if you are looking for a quick and easy access to all the technical information about a Siemens Industry product. For example, for comparing products, if you are expanding your system or replacing individual components, this is how to do it:

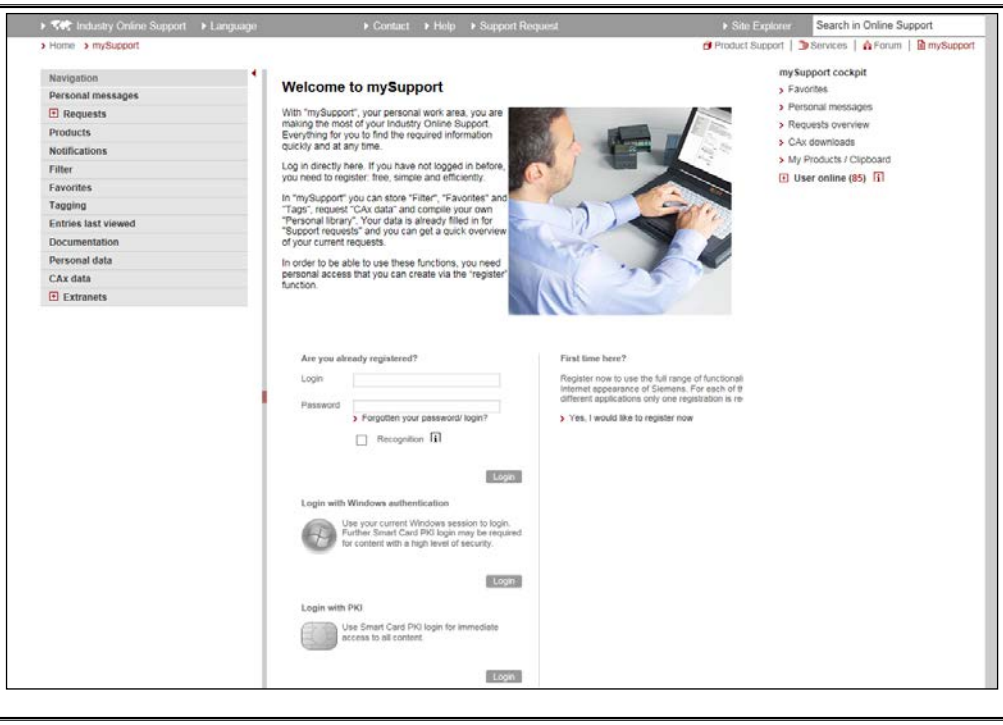
In the Product Support area, there is the central navigation bar.

To select a product, simply select the filter “Product.” Enter an order number or a product name here. You will be supported by a dynamic display of suitable products (list of suggestions).

One more click and the details of the selected product will be displayed – always up to date:

- Product life cycle, consisting of milestones with dates (e.g. delivery release, discontinuation of the product, …). You will find out whether the selected product is a current product or whether the product is already in the discontinuation phase.
- Successor products for discontinued products and new developments will be suggested. If there is a successor product, you will get a direct link to the product information of this product.
- Technical data – clear, compact and complete. You get all the available technical data concerning the selected product here – dimensions, operating voltage or the number of inputs/outputs, etc.

12.8. mySupport – Overview


Industry Online Support
Language
Contact
Help
Support Request
Site Explorer
Search in Online Support
Home
mySupport
Product Support
Services
Forum
mySupport
Navigation
Personal messages
Requests
Products
Notifications
Filter
Favorites
Tagging
Entries last viewed
Documentation
Personal data
CAx data
Extranets
Welcome to mySupport
With "mySupport", your personal work area, you are making the most of your Industry Online Support. Everything for you to find the required information quickly and at any time.
Log in directly here. If you have not logged in before, you need to register: free, simple and efficiently.
In "mySupport" you can store "Filter", "Favorites" and "Tags", request "CAx data" and compile your own "Personal library". Your data is already filled in for "Support requests" and you can get a quick overview of your current requests.
In order to be able to use these functions, you need personal access that you can create via the "register" function.
mySupport cockpit
Favorites
Personal messages
Requests overview
CAx downloads
My Products / Clipboard
User online (85)
Are you already registered?
Login
Password
Forgotten your password/ login?
Recognition
Login
Login with Windows authentication
Use your current Windows session to login. Further Smart Card PKI login may be required for content with a high level of security.
Login
Login with PKI
Use Smart Card PKI login for immediate access to all content.
Login
First time here?
Register now to use the full range of functionali
Internet appearance of Siemens. For each of t
different applications only one registration is re
Yes, I would like to register now

mySupport

The mySupport area will always remain your personal workplace; with this feature you can make the best of your Industry Online Support experience.

The most important thing, if you're already working with mySupport, you can take all your previous personal data and information you've filed away with you to the Industry Online Support.

In this area, you can compile the information that is important for your daily work – we provide you with the suitable tools. Create your own folder structures and file information such as bookmarks. There are numerous options, whether you want to file items by project or by products.

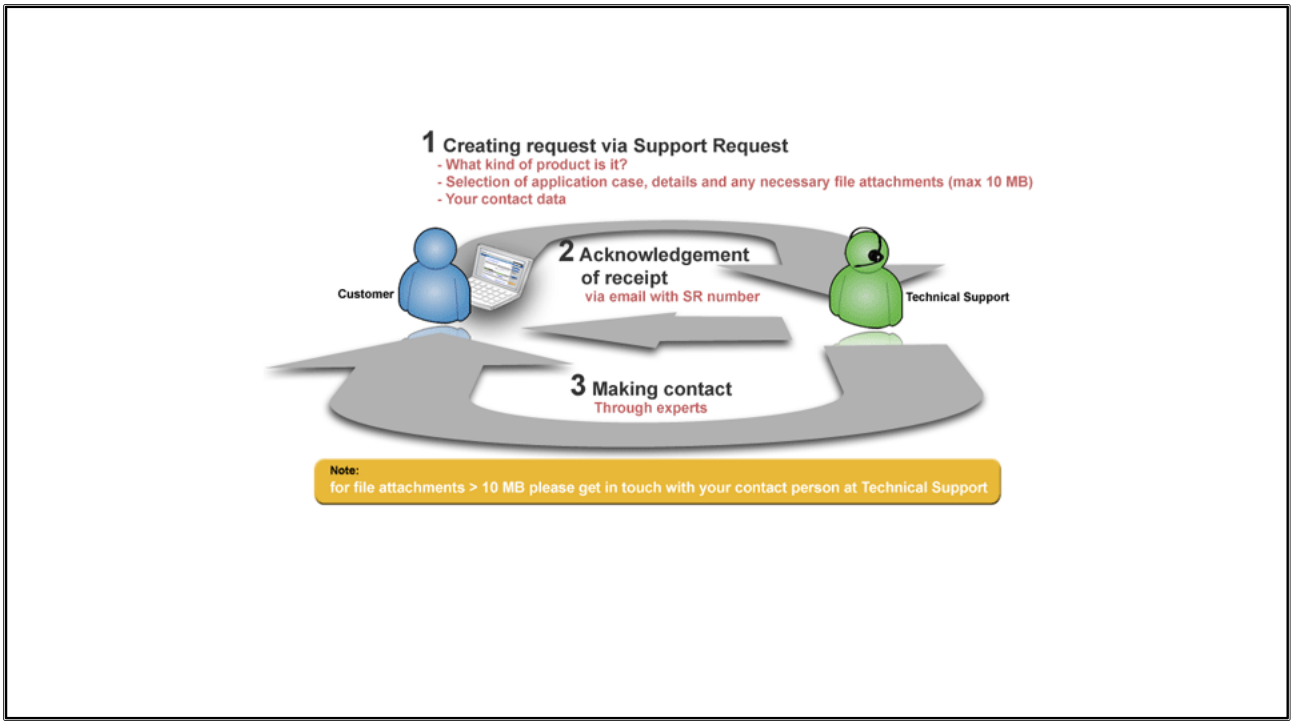
Moreover, you can now add notes, comments and tags (keywords). The system automatically creates a "Tag Cloud" based on your entries so you can access information quickly and easily by means of your own terms. The operation is consistent throughout mySupport so that you will easily find your way around. "Drag &drop" is also possible.

As soon as you are logged on, the mySupport cockpit is always at your side. It will immediately show you when the status of a support request changes, or when you receive new personal messages. You also have direct access to your personal keywords in the tag cloud, to the entries last visited, and you can see which user is online.

Here, just a few highlights:

- The previous MyDocumentationManager is now completely integrated into mySupport under the name of "mySupport-Documentation." The function category "Documentation" contains all the functions of the MyDocumentationManager and provides a few innovations, too.
- The Service & Support Newsletter has been completely revamped. An individual messaging system will more than replace it.

12.9. Support Request


1 Creating request via Support Request
- What kind of product is it?
- Selection of application case, details and any necessary file attachments (max 10 MB)
- Your contact data
Customer
2 Acknowledgement of receipt
via email with SR number
Technical Support
3 Making contact
Through experts
Note:
for file attachments > 10 MB please get in touch with your contact person at Technical Support

Support Request

To create a Support Request, different options are available to you in Online Support:

- You will find the "Support Request" option in the menu on all Online Support pages.
- Alternatively, you can create a new request in mySupport in the "Requests" category.
- Or directly click on the following link:

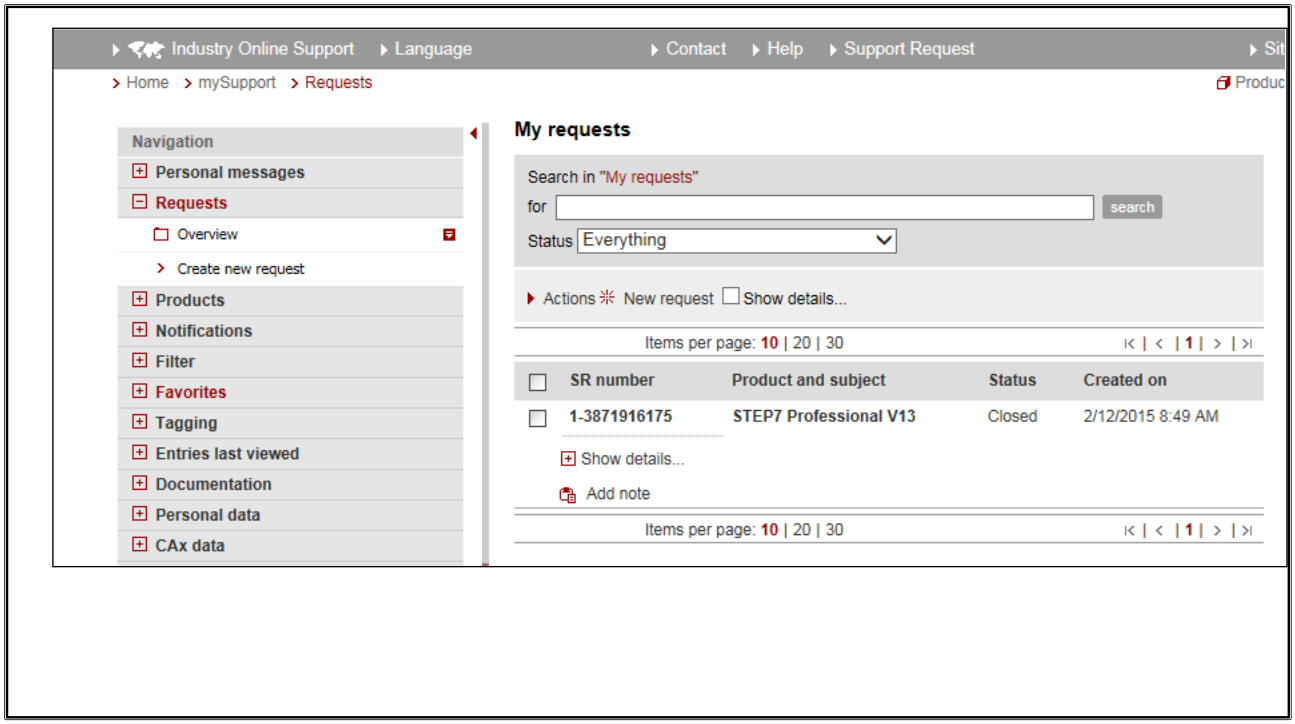
http://www.siemens.com/automation/support-request

Tips for creating a request:

- Select your product and use case as accurately as possible; try to avoid selecting "Other". By doing so, you ensure optimum support by our experts and appropriate suggested solutions.
- Did other users have a similar problem? This step already offers frequent problems and solutions. Take a look – it will be worth your while!
- Describe your problem with as much detail as possible. Pictures or explanatory attachments allow our experts to consider your problem from all sides and develop solutions. You can upload multiple attachments up to 10 MB per file.
- Before each sending, verify your personal contact information and the data you have entered. The final step additionally offers the option to print the summary.

As a logged in user, you can track the status of your requests online. To do so, navigate to "My requests" in the "Requests" category in mySupport.

12.10. Support Request



12.11. Industry Online Support – wherever you go

- Mobile access to more than 300,000 entries on all Siemens Industry products
- Reduced to the essential functions
- Application case: initial diagnosis of problem or in case of failures directly at the system or machine

Quick and easy access to technical information, anytime.
Scan your product codes, search and Support Request
fingertips at any time.

The app supports you, for example, in the following fields:

- Problem solving during the implementation of a project
- Troubleshooting of failures
- Expanding or restructuring your system

It also provides you with access to the Technical Forum and to further entries created for you by our experts:

- FAQs
- Application examples
- Manuals
- Certificates
- Product notes and many others

The main functions at a glance:

- Scan your product codes / EAN codes for a direct display of all technical and graphic data (e.g. CAx data) about your Siemens Industry product.
- Send your product information or entries per e-mail in order to process the information directly at the workstation.
- Send your requests to Technical Support at your convenience. Detail information can easily be added using the scan or photo function.
- Use the offline cache function to save your favorites to your device. In this way you can call these entries, products and conferences even without network coverage.
- Transfer PDF documents to an external library.
- The contents and surfaces are available in six languages (German, English, French, Italian, Spanish and Chinese) - including a temporary switching to English.

12.11.1. Scanning product / EAN code

- After scanning a code, the product view will open in most cases.
- Specialized search options are available for configurable products.
- Depending on the product, the result can also be displayed as a list of products, in which
- the user can select the appropriate product.

12.11.2. Scan functionality

Data matrix codes

on Siemens products
as per standard SN60450

QR code

e.g.: in advertisements
relating to Siemens content

EAN13 bar code

on Siemens products

SN60450

URL).

Code39 bar code

(very hard to recognize /
scan) on Siemens products

The scan functionality in the Online Support app supports the following types of code:

Data matrix code

QR code

EAN13 bar code

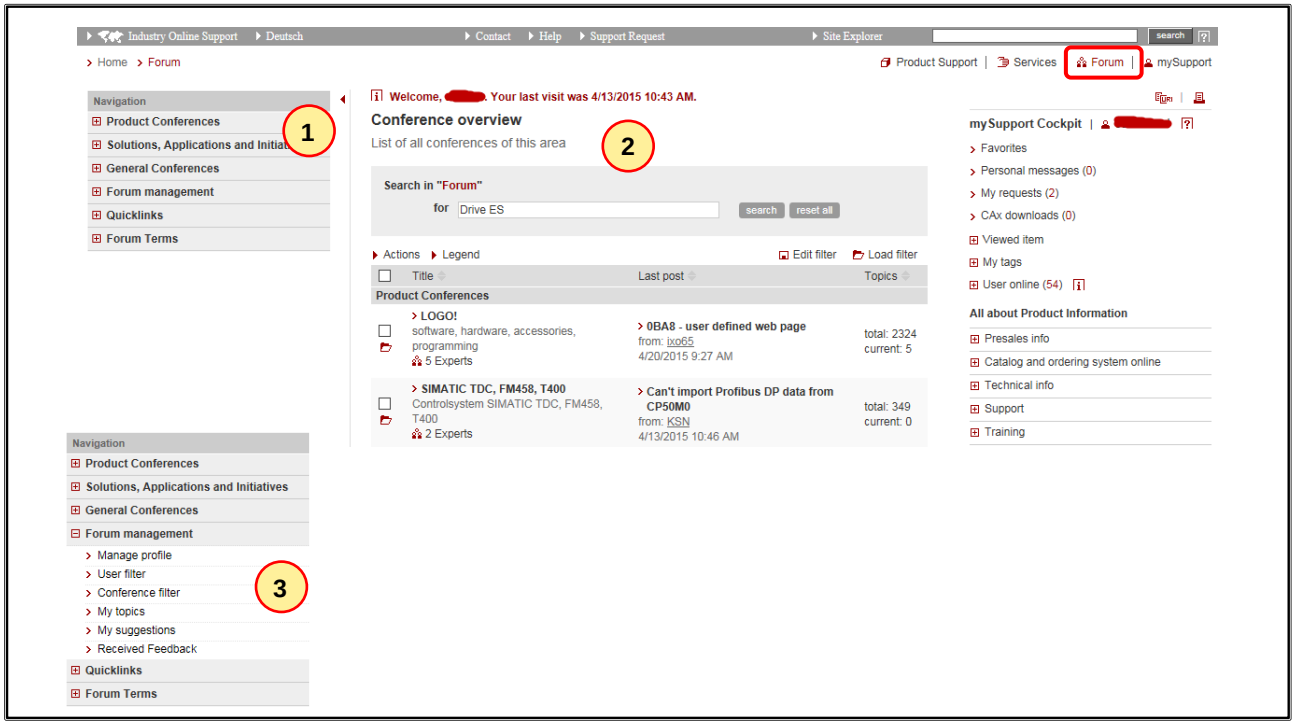
Code39 bar code

When one of these codes is recognized, the respective product view is called up in the app.

Exception:
The QR codes contain URLs – these are directly called up and displayed in the app by the integrated browser (but only, if "siemens" is contained in the

12.12. Forum - the communication platform for Siemens Industry products

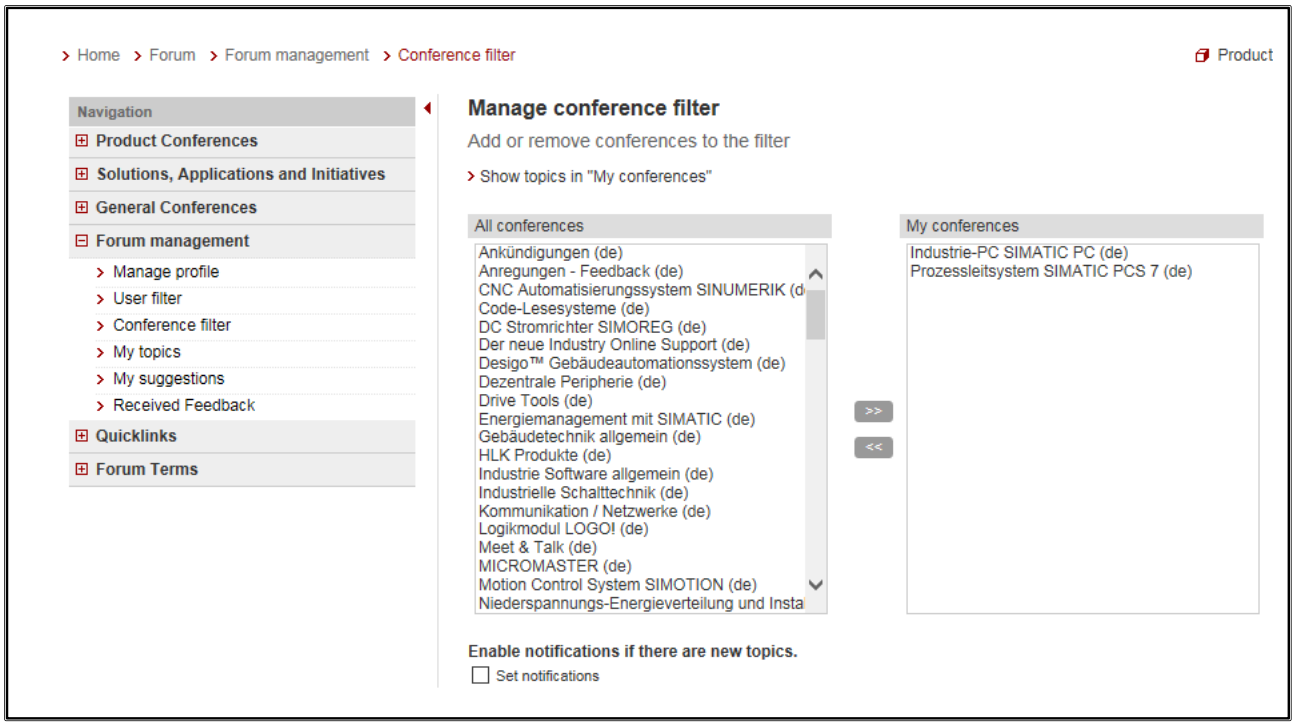
12.12.1. Conferences and Forum management



1 On the left side, you will find the so-called conference tree. It allows you to navigate through the individual discussion areas.

2 The conference overview is the central discussion area of the Technical Forum. This is where the community meets to discuss technical questions about Siemens Industry products.

3 In forum management, you will find your personal control center for the Technical Forum. It allows you to manage your specific profile data and filters.



Conference filter

Add conferences to your personal filter of preferred conferences.

This allows you to enable a notification that informs you when new topics are started in these conferences.

In Quicklinks, the Technical Forum additionally offers an overview page that contains all topics of your preferred conferences.

Managing profile

Profile management provides interesting information and functions:

- You get an overview of your activities in the Technical Forum.
- You can view your rank, any special permissions and your ranking progress.
- You can store a signature and a personal description for your profile in the forum.
- You have direct access to the quick links to get an overview of all topics you have contributed to.

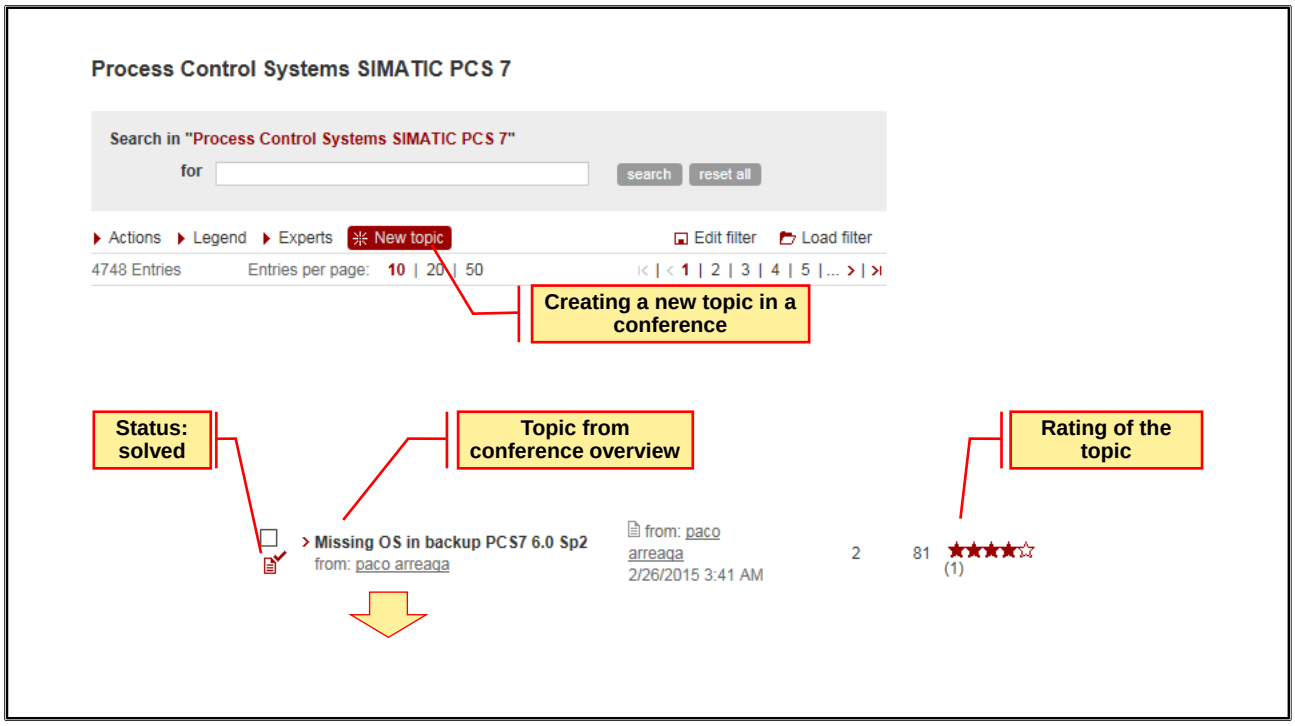
User filter

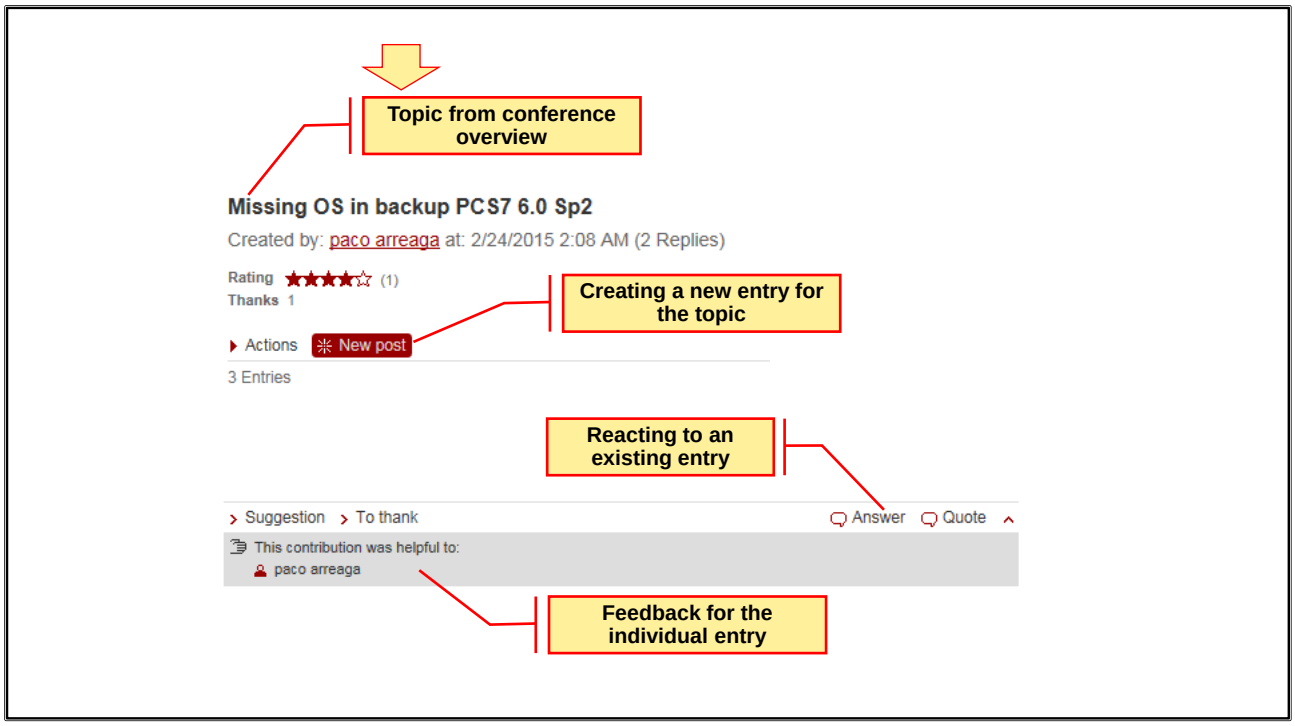
Have you found a user in the Technical Forum who posts entries that are particularly interesting? Then add this user to your list of "preferred users".

This allows you to enable a notification that informs you when the user has posted a new entry.

In Quicklinks, the Technical Forum additionally offers an overview page that contains all topics of your preferred users.

12.12.2. Interactions in the Forum





Creating a new entry

Do you want to create or format a new entry? The entry editor provides all the necessary functions.

- You can upload and publish in the forum a file with "Add attachment".
- You would like to check before the publication how your entry will actually look? A preview is available for this purpose.
- You would like to look at the topic again to which you create an entry? Please, you used the link over the input area (right mouse button > open in a new tab or window)

Posting / replying to an entry

Do you want to participate in an existing discussion with your own entry? Click on "Reply" and post your personal entry to support other users in answering the question.

- Use the "Reply" link to go to the entry editor and create a reply without quoting the entry.
- If you want to quote the entry, possibly only excerpts of it, use the "Quote" link. The content of the quoted entry is then displayed accordingly in the entry editor.

Rating an entry / saying thank you

Do you find an entry particularly interesting? Use the available functions and rate the entry or say thank you to provide personal feedback. Ratings and thank yours are the rewards our community members get for the support they provide. When you rate an author or entry, this will be added to the already existing ratings. The average value of all ratings is displayed.

Aside from feedback to the author of the entry, you also draw other readers' attention to particularly valuable entries and helpful authors.

12.13. Task and Checkpoint

Task: Software compatibility

Goal

Find out which current version of virus scanners is compatible with your engineering software.

Use all information sources available:

- Readme files in the installation folder
- The compatibility tool of the Industry Online Support
- Entries in the Product support
- Entries in the Forum
- Create a Support Request.

Checkpoint

Let‘s think about this:

- **Name some reasons for registration in MySupport.**
- **What do you think is the best way to have always the latest version of the required manuals for your job with you?**

Siemens AG
Digital Factory
SITRAIN – Digital Industry Academy
Postfach 48 48
90026 Nürnberg

siemens.com/sitrain